(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2018년04월27일
(11) 등록번호 10-1853222
(24) 등록일자 2018년04월23일

(51) 국제특허분류(Int. Cl.)
G06F 9/38 (2006.01) *G06F 9/30* (2018.01)
G06F 9/50 (2018.01)
(52) CPC특허분류
G06F 9/3851 (2013.01)
G06F 9/30145 (2013.01)
(21) 출원번호 10-2016-7023806
(22) 출원일자(국제) 2015년03월16일
심사청구일자 2016년08월29일
(85) 번역문제출일자 2016년08월29일
(65) 공개번호 10-2016-0114170
(43) 공개일자 2016년10월04일
(86) 국제출원번호 PCT/EP2015/055447
(87) 국제공개번호 WO 2015/144479
국제공개일자 2015년10월01일
(30) 우선권주장
14/226,947 2014년03월27일 미국(US)
(56) 선행기술조사문헌
US20080114973 A1*
US20040073779 A1*
US20040215932 A1
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
인터내셔널 비지네스 머신즈 코포레이션
미국 10504 뉴욕주 아몬크 뉴오차드 로드
(72) 발명자
그라이너, 댄
미국 캘리포니아 95141-1003, 산 호세, 산타 테레사 랩, 555 베일리 애비뉴, 메일 드롭 Sv1/090/f374, 아이비엠 코포레이션
파렐, 마크
미국 뉴욕 12601-5400, 포킵시, 사우스 로드 2455, 메일 드롭 P310, 아이비엠 코포레이션
(뒷면에 계속)
(74) 대리인
허정훈

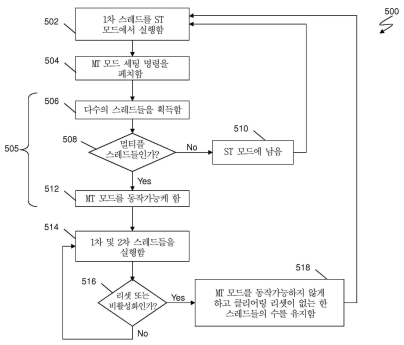
전체 청구항 수 : 총 8 항 심사관 : 서광훈

(54) 발명의 명칭 **멀티스레딩 컴퓨터 시스템에서 어드레스 확장 및 수축**

(57) 요 약

컴퓨터 시스템은 단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에 구성 가능한 코어를 갖는 구성을 포함한다. 상기 ST 모드는 1차 스레드를 어드레스하고, 상기 MT 모드는 1차 스레드 및 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스한다. 코어 어드레스 값을 사용하여 상기 ST 모드에서 1차 스레드를 액세스하고 상기 ST 모드로부터 상기 MT 모드로 전환하는 단계(switching)를 포함하는 방법을 수행하기 위해 상기 구성의 이용을 컨트롤하도록 멀티스레딩 퍼실리티(a multithreading facility)가 구성된다. 상기 1차 스레드 또는 상기 하나 또는 그 이상의 2차 스레드들 중 하나는 확장된 어드레스 값을 사용하여 상기 MT 모드에서 액세스되고, 상기 확장된 어드레스 값은 스레드 어드레스 값과 연결된 상기 코어 어드레스 값을 포함한다.

대 표 도



(52) CPC특허분류
G06F 9/5077 (2013.01)
(72) 발명자
오시섹, 다미안, 레오
미국 뉴욕 13760-5553, 엔디코트, 아이비엠 노스 스트리트 1701, 메일 드롭 G28g/250-2, 아이비엠 코포레이션
슈미트, 도날드, 윌리엄
미국 뉴욕 12601-5400, 포킵시, 사우스 로드 2455, 메일 드롭 7q4a/p334, 아이비엠 코포레이션
부사바, 파디, 유섭
미국 뉴욕 12601-5400, 포킵시, 사우스 로드 2455, 메일 드롭 P312, 아이비엠 코포레이션
쿠발라, 제프리, 폴
미국 뉴욕 12601-5400, 포킵시, 사우스 로드 2455, 메일 드롭 P340, 아이비엠 코포레이션
브래드버리, 조나단, 데이비드
미국 뉴욕 12601-5400, 포킵시, 사우스 로드 2455, 아이비엠 코포레이션
헬러, 리사, 크랜튼
미국 뉴욕 12601-5400, 포킵시, 사우스 로드 2455, 메일 드롭 A85/p310, 아이비엠 코포레이션
슬레겔, 티모시
미국 뉴욕 12601-5400, 포킵시, 사우스 로드 2455, 메일 드롭 Ms-p310, 아이비엠 코포레이션

명 세 서

청구범위

청구항 1

컴퓨터 시스템에서, 상기 시스템은:

단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에서 구성 가능한 코어(a core configurable)를 포함하는 구성(a configuration) - 상기 ST 모드는 1차 스레드(a primary thread)를 어드레스하고(addressing) 그리고 상기 MT 모드는 상기 1차 스레드와 상기 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스함 -; 및

방법을 수행하기 위해 상기 구성에 대한 이용(utilization)을 컨트롤하도록 구성된 멀티스레딩 퍼실리티(a multithreading facility)를 포함하고, 상기 방법은:

코어 어드레스 값(a core address value)을 사용하여 상기 ST 모드에서 1차 스레드를 액세스하는 단계(accessing);

상기 ST 모드로부터 상기 MT 모드로 전환하는 단계(switching);

요청된 최대 스레드 식별자에 기초하는 양 만큼(by an amount based on a requested maximum thread identifier) 시프트함으로써 시프트된 코어 어드레스 값(a shifted core address value)으로서 확장된 어드레스 값(an expanded address value)을 형성하고(forming) 상기 시프트된 코어 어드레스를 스레드 어드레스 값과 연결하는 단계(concatenating);

상기 확장된 어드레스 값(an expanded address value)을 사용하여 상기 1차 스레드 또는 상기 MT 모드에서 상기 하나 또는 그 이상의 2차 스레드들 중 하나를 액세스하는 단계(accessing);

상기 MT 모드와 상기 ST 모드 사이에서 전환하는 단계(switching);

상기 코어가 상기 ST 모드에 있음에 기초하여 상기 1차 스레드를 액세스하기 위해 상기 코어 어드레스 값을 선택하는 단계(selecting); 및

상기 1차 스레드 또는 상기 코어가 상기 MT 모드에 있음에 기초하여 상기 하나 또는 그 이상의 2차 스레드 중 하나를 액세스하기 위해 상기 확장된 어드레스 값을 선택하는 단계(selecting)를 포함하는

컴퓨터 시스템.

청구항 2

삭제

청구항 3

삭제

청구항 4

삭제

청구항 5

제1항에 있어서, 표준-형식 어드레스(a standard-format address)가 상기 ST 모드에서 사용되고, 상기 코어는 상기 MT 모드가 동작 가능하지 않게 되는 것(disabling)에 기초하여 상기 MT 모드로부터 상기 ST 모드로 복귀하는(revert)

컴퓨터 시스템.

청구항 6

제5항에 있어서, 상기 MT 모드가 동작 가능하지 않게 되는 것(disabling)에 기초하여 오직 상기 1차 스레드만 액세스 가능하게 되고, 상기 하나 또는 그 이상의 2차 스레드들은 액세스 가능하지 않게 되는

컴퓨터 시스템.

청구항 7

삭제

청구항 8

삭제

청구항 9

단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에서 구성 가능한 코어(a core configurable)를 포함하는 구성(a configuration) - 상기 ST 모드는 1차 스레드(a primary thread)를 어드레스하고(addressing) 그리고 상기 MT 모드는 상기 1차 스레드와 상기 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스함 - 에서 어드레스 조정(address adjustment)을 위한 컴퓨터 구현 방법에서, 상기 방법은:

코어 어드레스 값(a core address value)을 사용하여 상기 ST 모드에서 1차 스레드를 액세스하는 단계(accessing);

상기 ST 모드로부터 상기 MT 모드로 전환하는 단계(switching);

요청된 최대 스레드 식별자에 기초하는 양 만큼(by an amount based on a requested maximum thread identifier) 시프트함으로써 시프트된 코어 어드레스 값(a shifted core address value)으로서 확장된 어드레스 값(an expanded address value)을 형성하고(forming) 상기 시프트된 코어 어드레스를 스레드 어드레스 값과 연결하는 단계(concatenating);

상기 확장된 어드레스 값(an expanded address value)을 사용하여 상기 1차 스레드 또는 상기 MT 모드에서 상기 하나 또는 그 이상의 2차 스레드들 중 하나를 액세스하는 단계(accessing);

상기 MT 모드와 상기 ST 모드 사이에서 전환하는 단계(switching);

상기 코어가 상기 ST 모드에 있음에 기초하여 상기 1차 스레드를 액세스하기 위해 상기 코어 어드레스 값을 선택하는 단계(selecting); 및

상기 1차 스레드 또는 상기 코어가 상기 MT 모드에 있음에 기초하여 상기 하나 또는 그 이상의 2차 스레드 중 하나를 액세스하기 위해 상기 확장된 어드레스 값을 선택하는 단계(selecting)를 포함하는

컴퓨터 구현 방법.

청구항 10

삭제

청구항 11

삭제

청구항 12

삭제

청구항 13

제9항에 있어서, 표준-형식 어드레스(a standard-format address)가 상기 ST 모드에서 사용되고, 상기 코어는 상기 MT 모드가 동작 가능하지 않게 되는 것(disabling)에 기초하여 상기 MT 모드로부터 상기 ST 모드로 복귀하는(revert)

컴퓨터 구현 방법.

청구항 14

제13항에 있어서, 상기 MT 모드가 동작 가능하지 않게 되는 것(disabling)에 기초하여 오직 상기 1차 스레드만 액세스 가능하게 되고, 상기 하나 또는 그 이상의 2차 스레드들은 액세스 가능하지 않게 되는

컴퓨터 구현 방법.

청구항 15

삭제

청구항 16

단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에서 구성 가능한 코어(a core configurable)를 포함하는 구성(a configuration) - 상기 ST 모드는 1차 스레드(a primary thread)를 어드레스하고(addressing) 그리고 상기 MT 모드는 상기 1차 스레드와 상기 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스함 - 에서 어드레스 조정을 위한 방법을 수행하는, 컴퓨터 시스템에 의해서 판독 가능한, 프로그램 명령들이 구현되어 있는 컴퓨터 판독가능 스토리지 매체 - 상기 컴퓨터 판독가능 스토리지 매체는 신호가 아님 - 에 있어서, 상기 방법은:

코어 어드레스 값(a core address value)을 사용하여 상기 ST 모드에서 1차 스레드를 액세스하는 단계(accessing);

상기 ST 모드로부터 상기 MT 모드로 전환하는 단계(switching);

요청된 최대 스레드 식별자에 기초하는 양 만큼(by an amount based on a requested maximum thread identifier) 시프트함으로써 시프트된 코어 어드레스 값(a shifted core address value)으로서 확장된 어드레스 값(an expanded address value)을 형성하고(forming) 상기 시프트된 코어 어드레스를 스레드 어드레스 값과 연결하는 단계(concatenating);

상기 확장된 어드레스 값(an expanded address value)을 사용하여 상기 1차 스레드 또는 상기 MT 모드에서 상기 하나 또는 그 이상의 2차 스레드들 중 하나를 액세스하는 단계(accessing);

상기 MT 모드와 상기 ST 모드 사이에서 전환하는 단계(switching);

상기 코어가 상기 ST 모드에 있음에 기초하여 상기 1차 스레드를 액세스하기 위해 상기 코어 어드레스 값을 선택하는 단계(selecting); 및

상기 1차 스레드 또는 상기 코어가 상기 MT 모드에 있음에 기초하여 상기 하나 또는 그 이상의 2차 스레드 중 하나를 액세스하기 위해 상기 확장된 어드레스 값을 선택하는 단계(selecting)를 포함하는

컴퓨터 판독가능 스토리지 매체.

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

제16항에 있어서, 표준-형식 어드레스(a standard-format address)가 상기 ST 모드에서 사용되고, 상기 코어는 상기 MT 모드가 동작 가능하지 않게 되는 것(disabling)에 기초하여 상기 MT 모드로부터 상기 ST 모드로 복귀하며, 상기 MT 모드가 동작 가능하지 않게 되는 것(disabling)에 기초하여 오직 상기 1차 스레드만 액세스 가능하게 되고, 상기 하나 또는 그 이상의 2차 스레드들은 액세스 가능하지 않게 되는

컴퓨터 판독가능 스토리지 매체.

발명의 설명

기 술 분 야

[0001] [0001] 본 발명은 일반적으로 다수 스레드들(multiple threads)을 지원하는 컴퓨터 시스템에 관한 것이고, 더 구체적으로는 멀티스레딩(multithreading) 컴퓨터 시스템에서 어드레스 확장 및 수축(address expansion and contraction)에 관한 것이다.

배 경 기 술

[0002] [0002] 컴퓨터 시스템들의 프로세서 속도는 과거에 비해 증가되었지만 그러한 컴퓨터 시스템들의 메모리에 액세스할 수 있는 속도는 비례해서 증가되지 않았다. 따라서, 프로세스의 사이클 타임이 빨라질수록 메모리로부터 페치될 데이터를 기다리는 지연은 더 심화되었다. 그러한 지연들의 효과들은 다양한 수준들의 캐싱으로, 최근의 프로세서들에서는, 멀티스레딩(MT)에 의해서 완화되어 왔다.

[0003] [0003] MT는 프로세서의 다양한 코어 자원들이 스레드들(threads)로 알려진 복수의 명령 스트림들에 의해서 공유되도록 해준다. 코어 자원들은 실행 유닛들, 캐시들, 변환-룩어사이드 버퍼들(TLB들), 및 이와 유사한 것들을 포함할 수 있고, 이들은 집단적으로 가리켜 일반적으로 코어라 한다. 한 스레드에서의 캐시-미스 또는 기타 지연으로 인한 대기 동안, 하나 또는 그 이상의 다른 스레드들이 코어 자원들을 이용할 수 있고, 따라서 코어 자원들의 이용률을 증가시킬 수 있다. 수퍼-스칼라 프로세서 동시-멀티스레딩(SMT) 구현(a super-scalar processor simultaneous-multithreading (SMT) implementation)에서, 다수의 스레드들은 하나 또는 그 이상의 코어들의 코어 자원들에 의해서 동시에 서비스될 수 있다.

[0004] [0004] 현대의 하드웨어 플랫폼들에서, MT는 통상적으로 MT 하드웨어 상에서 동작하는 운영 체제(OS)에 대하여 투명한 방식으로 구현된다. 이 특성의 한 특징은 MT 하드웨어를 이용하기 위해 OS가 수정을 요구하지 않는다는 것이다. 그러나, OS에 관한 투명한 MT 연산은 응답 시간(response time), 용량 프로비저닝(capacity provisioning), 용량 플래닝(capacity planning) 및 빌링(billing)의 높은 변동성을 초래할 수 있다. 이 변동성은 OS가 자신의 작업들이 코어의 배타적 컨트롤을 갖고 있는지 또는 자신의 작업들이 한 코어를 공유하는 스레드들로서 실행되고 있는지 알지 못하기 때문에 발생할 수 있다. 설계에 의해서, MT-가능한 하드웨어 상의 메모리-집중적인 워크로드를 위한 가장 높은 용량(the highest capacity for a memory-intensive workload)이 달성 가능한데, 코어들의 사용 중에 스레드 밀도(thread density)가 높은 평균(high average)일 때 달성 가능하다. MT에 의해서 제공되는 캐시 이용(cache exploitation)이 증가하면 추가의 용량이 필요할 수 있다. 만일 OS가 이용되는 코어들에 대해 높은 평균 스레드 밀도를 일관되게 유지하지 않는다면, MT에 의해서 제공된 추가의 전체 스루풋 용량(the additional overall throughput capacity)은 이용 가능하지 않게 된다. 예를 들어, 만일 하드웨어가 계산 이용률(compute utilization)이 낮을 때 코어 당 단일 MT 스레드를 실행하고 계산 이용률이 높을 때 높은 스레드 밀도로 실행한다면, 얼마나 많은 전체 MT 계산 용량이 워크로드에 대해서 이용 가능한지를 결정하는 것은 매우 어려울 수 있다. MT 스레드 이용에서 이러한 하드웨어 변동성은 용량에 관해서 전술한 바와 같은 비슷한 방식으로 트랜잭션 응답 시간들 및 빌링 모두에서의 변동성이 될 수 있다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0005] [0005] 실시 예들은 멀티스레딩 컴퓨터 시스템에서 어드레스 확장 및 수축을 위한 시스템, 방법, 및 컴퓨터 프로그램 제품을 포함한다. 한 실시 예에 따라, 컴퓨터 시스템은 단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에서 구성 가능한 코어(a core configurable)를 갖는 구성(a configuration)을 포함한다. 상기 ST 모드는 1차 스레드(a primary thread)를 어드레스하고(address) 그리고 상기 MT 모드는 상기 1차 스레드와 상기 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스한다. 코어 어드레스 값(a core address value)을 사용하여 상기 ST 모드에서 1차 스레드를 액세스하고 상기 ST 모드로부터 상기 MT 모드로 전환하는 단계

(switching)를 포함하는 방법을 수행하기 위해 상기 구성의 이용을 컨트롤하도록 멀티스레딩 퍼실리티(a multithreading facility)가 구성된다. 상기 1차 스레드 또는 상기 하나 또는 그 이상의 2차 스레드들 중 하나는 확장된 어드레스(an expanded address value)를 사용하여 상기 MT 모드에서 액세스되고, 상기 확장된 어드레스 값은 스레드 어드레스 값과 연결된(concatenated) 상기 코어 어드레스 값을 포함한다.

[0006] [0006] 다른 실시 예에 따라, 구성에서 어드레스 조정(address adjustment in a configuration)을 위한 컴퓨터 구현 방법(a computer implemented method)이 제공된다. 상기 구성은 단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에서 구성 가능한 코어(a core configurable)를 포함하고, 상기 ST 모드는 1차 스레드(a primary thread)를 어드레스하며(address) 그리고 상기 MT 모드는 상기 1차 스레드와 상기 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스한다. 상기 1차 스레드는 코어 어드레스 값(a core address value)을 사용하여 상기 ST 모드에서 액세스된다. 상기 ST 모드로부터 상기 MT 모드로 전환하는 단계(switching)가 수행된다. 상기 1차 스레드 또는 상기 하나 또는 그 이상의 2차 스레드들 중 하나는 확장된 어드레스(an expanded address value)를 사용하여 상기 MT 모드에서 액세스된다. 상기 확장된 어드레스 값은 스레드 어드레스 값과 연결된(concatenated) 상기 코어 어드레스 값을 포함한다.

[0007] [0007] 또 다른 실시 예는 구성에서 어드레스 조정(address adjustment in a configuration)을 위한 컴퓨터 프로그램 제품을 포함한다. 상기 구성은 단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에서 구성 가능한 코어(a core configurable)를 포함하고, 상기 ST 모드는 1차 스레드(a primary thread)를 어드레스하며(address) 그리고 상기 MT 모드는 상기 1차 스레드와 상기 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스한다. 상기 컴퓨터 프로그램 제품은 컴퓨터 판독가능 스토리지 매체를 포함하고, 상기 컴퓨터 판독가능 스토리지 매체는 신호가 아니며, 상기 컴퓨터 판독가능 스토리지 매체에는 프로그램 명령들이 구현되어 있다. 상기 프로그램 명령들은 처리 회로가 코어 어드레스 값(a core address value)을 사용하여 상기 ST 모드에서 1차 스레드를 액세스하고 상기 ST 모드로부터 상기 MT 모드로 전환하는 단계(switching)를 포함하는 방법을 수행하도록 상기 처리 회로에 의해서 판독가능하다. 상기 1차 스레드 또는 상기 하나 또는 그 이상의 2차 스레드들 중 하나는 확장된 어드레스(an expanded address value)를 사용하여 상기 MT 모드에서 액세스되고, 상기 확장된 어드레스 값은 스레드 어드레스 값과 연결된(concatenated) 상기 코어 어드레스 값을 포함한다.

도면의 간단한 설명

[0008] [0008] 실시 예들로서 간주되는 주제는 본 명세서의 결론 부분에 청구항들로 구체적으로 명확하게 기재되고 청구된다. 본 발명의 전술한 그리고 기타의 특징들, 및 장점들은 첨부된 도면들을 참조하여 작성된 아래의 상세한 설명으로부터 명백하다.

도 1a는 한 실시 예에 따라 구현될 수 있는 컴퓨팅 환경을 도시한다.

도 1b는 한 실시 예에 따라 구현될 수 있는 컴퓨팅 환경을 도시한다.

도 2는 한 실시 예에 따라 구현될 수 있는 코어의 처리 회로를 도시한다.

도 3은 한 실시 예에 따라 구현될 수 있는 컴퓨팅 환경을 도시한다.

도 4는 한 실시 예에 따라 구현될 수 있는 컴퓨팅 환경에서 하이퍼바이저 컨텍스트 유지의 한 예를 도시한다.

도 5는 한 실시 예에 따라 멀티스레딩의 동적 수행을 가능하게 하기 위한 처리 흐름을 도시한다.

도 6a는 한 실시 예에 따라 CPU 어드레스 확장 처리의 한 예를 도시한다.

도 6B는 한 실시 예에 따라 CPU 어드레스 수축 처리의 한 예를 도시한다.

도 7은 한 실시 예에 따라 세트-멀티스레딩 지시를 위한 처리 흐름을 도시한다.

도 8은 한 실시 예에 따라 멀티스레딩 능력 정보를 저장하는 것의 한 예를 도시한다.

도 9는 한 실시 예에 따라 멀티스레딩 능력을 결정하기 위한 처리 흐름을 도시한다.

도 10은 한 실시 예에 따라 다양한 스레드 컨텍스트 위치들의 한 예를 도시한다.

도 11은 한 실시 예에 따라 멀티스레딩 레지스터 보존의 한 예를 도시한다.

도 12는 한 실시 예에 따라 멀티스레딩 레지스터 보존을 위한 처리 흐름을 도시한다.

도 13은 한 실시 예에 따라 멀티스레딩 레지스터 복원의 한 예를 도시한다.

도 14는 한 실시 예에 따라 멀티스레딩 레지스터 복원을 위한 처리 흐름을 도시한다.

도 15는 한 실시 예에 따라 컴퓨터-판독가능 매체를 도시한다.

발명을 실시하기 위한 구체적인 내용

[0009] [0009] 예시적 실시 예들은 연산의 단일 스레드 및 멀티스레딩 모드를 지원하는 컴퓨터 시스템에서 멀티스레딩 연산을 제공한다. 여기서 사용된 바와 같이, 논리 스레드(a logical thread)는 단일 명령 스트림 및 그의 관련 상태를 지칭한다. 다시 말하면, 아키텍처 수준에서, 각각의 논리 스레드는 독립된 중앙 처리 유닛(CPU) 또는 프로세서를 나타낸다. 하드웨어 수준에서, 스레드는, 파견되었을 때(dispatched), 그 게스트 상태의 유지와 조합된, 논리 스레드와 관련된 명령 스트림의 실행이다. 따라서, "스레드"와 "CPU"라는 용어들은 여기서 서로 교환 가능하게 사용될 수 있다.

[0010] [0010] 한 예시적 실시 예에서, CPU는 명령 실행(instruction execution), 인터럽션 액션(interruption action), 타이밍 기능들(timing functions), 초기 프로그램 로딩(initial program loading), 및 기타 머신-관련 기능들(other machine-related functions)을 위해 시퀀싱 및 처리 퍼실리티들(sequencing and processing facilities)을 포함한다. CPU는 다양한 하부 물리적 구현들(a variety of underlying physical implementations)로 맵핑할 수 있는 논리 기능들을 정의한다. 명령들을 실행함에 있어서, CPU는 고정 길이의 이진 정수들 및 부동 소수점 수들(예를 들어, 이진수, 십진수, 십육진수), 가변 길이의 십진 정수들, 및 고정 또는 가변 길이의 논리 정보를 처리할 수 있다. 처리는 병렬로 또는 직렬로 수행될 수 있다. 처리 엘리먼트들의 폭(the width of processing elements), 시프팅 경로들의 다양성(multiplicity of shifting paths), 및 다른 종류의 산술 연산의 수행에서 동시성의 정도(the degree of simultaneity)는 CPU 모델에 따라 다를 수 있지만 논리 결과들에는 영향이 없다.

[0011] [0011] CPU가 실행하는 명령들은 일반(general), 십진(decimal), 부동-소수점-지원(floating-point-support (FPS)), 이진-부동-소수점(binary-floating-point (BFP)), 십진-부동-소수점(decimal-floating-point (DFP)), 십육진-부동-소수점(hexadecimal-floating-point (HFP)), 컨트롤(control), 및 I/O 명령들과 같은 다수의 명령의 종류들(classes)을 포함할 수 있다. 일반 명령들은 이진-정수-산술-연산들(binary-integer-arithmetic operations), 및 논리(logical) 연산들, 분기(branching) 연산들, 및 기타 비-산술 연산들(other non-arithmetic operations)을 수행하는데 사용될 수 있다. 십진 명령들은 십진 형식의 데이터에 관하여 연산한다. BFP, DFP, 및 HFP 명령들은 BFP, DFP, 및 HFP 형식의 데이터에 관하여, 각각, 연산하고, 한편 FPS 명령들은 형식에 무관하게 부동-소수점 데이터에 관하여 연산하거나 또는 한 형식에서 다른 형식으로 변환한다. 특권 컨트롤 명령들(privileged control instructions) 및 I/O 명령들은 CPU가 수퍼바이저 상태(a supervisor state)에 있을 때 실행될 수 있고, 반-특권 컨트롤 명령들(semi-privileged control instructions)은, 적절한 승인 메커니즘의 지휘 아래(subject to appropriate authorization mechanisms), 문제 상태(a problem state)에서 실행될 수 있다.

[0012] [0012] CPU는 프로그램들에 이용 가능한 레지스터들을 제공하지만 주 스토리지에서 어드레스할 수 있는 표현들(addressable representations)은 가지고 있지 않다. 레지스터들은, 예를 들어, 현재 프로그램-상태-워드(a current program-status word (PSW)), 일반 레지스터들(general registers), 부동-소수점 레지스터들 및 부동-소수점-컨트롤 레지스터(floating-point registers and a floating-point-control register), 벡터 레지스터들(vector registers), 컨트롤 레지스터들(control registers), 액세스 레지스터들(access registers), 프리픽스 레지스터들(a prefix register), 타임-오브-데이(TOD) 프로그램 가능 레지스터(a time-of-day (TOD)-programmable register), 그리고 클럭 비교기 및 CPU 타이머를 위한 레지스터(registers for a clock comparator and CPU timer)를 포함할 수 있다. 이러한 레지스터들의 세트를 CPU의 아키텍처된 레지스터 컨텍스트(the CPU's architected register context)라 한다. 구성에서 각각의 CPU는 TOD 클럭에 대해 액세스를 제공할 수 있고, 이 TOD 클럭은 상기 구성의 모든 CPU들에 의해서 공유될 수 있다. 명령 연산 코드는 연산에서 어떤 종류의 레지스터가 사용되어야 하는지를 결정할 수 있다.

[0013] [0013] 각각의 CPU는 타입 속성(a type attribute)을 가질 수 있는데, 이는 CPU가 기능들 및 퍼실리티들의 전체(a full complement of functions and facilities)를 제공하는지((예를 들어, 범용 CPU인지), 또는 CPU가 특정 타입의 워크로드들을 처리하도록 의도된 것인지(예를 들어, 특별 CPU인지)를 표시한다. 1차 CPU(primary CPU)는 일반 CPU이거나 또는 최종 초기 프로그램 로드(IPL) 연산(a last initial program load (IPL) operation) 이후

시작된 CPU (IPL CPU)와 동일 타입을 갖는 CPU이다. 2차 CPU(secondary CPU)는 IPL CPU와 다른 CPU 타입을 갖는 일반 CPU가 아닌 모든 CPU이다.

[0014] [0014] 멀티스레딩 퍼실리티는 지원 아키텍처(a supporting architecture)를 구현하는 컴퓨터 시스템 상에서 이용 가능할 수 있다. 멀티스레딩 퍼실리티는 코어를 공유하는, 한 그룹의 스레드들 – 이들은 또한 CPU들이라고도 한다 – 을 동작 가능하게 하는 멀티스레딩을 지원한다. 멀티스레딩 퍼실리티가 동작 가능하게 될 때, 코어 내의 CPU들은 실행 유닛들 또는 캐시들과 같은, 특정 하드웨어 자원들을 공유할 수 있다. 코어 내의 한 CPU가 하드웨어 자원들을 기다리며 대기할 때(통상적으로, 메모리 액세스를 기다림), 코어 내의 다른 CPU들은 그 코어 내의 공유 자원들을 사용하지 않고 놓아두기(remain idle)보다는 사용할 수 있다. 멀티스레딩 퍼실리티가 설치되고 동작 가능하게 될 때, 스레드는 코어의 멤버인 CPU를 뜻한다(synonymous). 멀티스레딩 퍼실리티가 설치되지 않았거나, 또는 멀티스레딩 퍼실리티가 설치는 되었지만 동작 가능하게 되지는 않았을 때, 코어는 단일 CPU 또는 스레드를 포함한다.

[0015] [0015] 멀티스레딩 퍼실리티가 설치되었을 때, 그것은 세트-멀티스레딩 신호 프로세서(SIGP) 지시(a set-multithreading signal processor (SIGP) order)의 실행에 의해서 동작 가능하게 될 수 있다. 한 예시적 실시예에서, 멀티스레딩 퍼실리티가 동작 가능하게 될 때, 구성 내의 CPU들의 수는 다수로 증가되는데, 그 값은 프로그램-명시된 최대 스레드 식별 정보(a program-specified maximum thread identification (PSMTID))에 의해서 결정된다. 한 코어 내의 CPU의 수는 PSMTID 이상일 수 있다. 이 다수에 대응하는 다수의 CPU들이 한 코어 내로 그룹화된다. 구성 내에서 동일 CPU 타입의 각 코어는 동일 개수의 CPU들을 가질 수 있다. 코어 내의 각각의 CPU는 동일 CPU 타입을 갖지만; 그러나, 모델과 CPU 타입에 기초하여, 코어 내의 일부 CPU들은 동작 가능하지 않을 수 있다.

[0016] [0016] 한 예시적 실시 예에서, 운영 체제(OS)와 같은, 컨트롤 프로그램은 OS가 관리하는 구성이 멀티스레딩을 사용할 수 있도록 멀티스레딩을 명시적으로(explicitly) 동작 가능하게 한다. 이와 달리, 하이퍼바이저(a hypervisor)도 멀티스레딩을 동작 가능하게 할 수 있고, 하이퍼바이저의 게스트들 및 그들의 애플리케이션들도 투명하게 혜택을 누릴 수 있다. 애플리케이션 프로그램은 일반적으로 멀티스레딩이 동작 가능하게 되었는지를 알지 못한다. 멀티스레딩이 동작 가능하게 되었을 때, 구성 내의 모든 CPU들의 CPU 어드레스들은 어드레스의 최좌측 비트들(the leftmost bits) 내에 코어 식별 정보(또는 코어 ID)를 그리고 어드레스의 최우측 비트들(the rightmost bits) 내에 스레드 식별 정보(스레드 ID 또는 TID)를 포함하도록 조정된다. 코어 ID는 또한 코어 어드레스 값이라고도 하고, TID는 스레드 어드레스 값이라 한다. 코어 내의 CPU들은 실행 유닛들 또는 하위-수준 캐시들(lower-level caches)과 같은 특정 하드웨어 퍼실리티들을 공유할 수 있고, 따라서, 코어의 한 CPU 내의 실행은 코어 내의 다른 CPU들의 수행에도 영향을 끼칠 수 있다.

[0017] [0017] 단일 스레드와 멀티스레딩 모드들 사이에서 한 구성의 하나 또는 그 이상의 코어들을 동적으로 전환하는 것과 관련된 변경들을 관리하기 위해서, 다수의 지원 특징들(a number of support features)이 포함된다. 멀티스레딩을 지원하지 않는 프로그램들과의 호환성을 유지하기 위해, 단일 스레드 모드는 리셋 또는 비활성화 시에(upon a reset or deactivation) 디폴트 모드(default mode)가 될 수 있다. 예시적 실시 예들은 멀티스레딩 모드에서 단일 스레드 모드로 전환한 후에 스레드 컨텍스트의 분석 및/또는 복원을 지원하기 위해 멀티스레딩 모드로부터 스레드 컨텍스트를 보존(preserve), 통신(communicate), 및 복원(restore)하기 위한 특징들을 포함한다.

[0018] [0018] 예시적 실시 예에 의해서 구현될 수 있는 컴퓨팅 환경은, 예를 들어, 미국 뉴욕주 아몬크에 소재하는 인터내셔널 비지네스 머신즈 코포레이션(IBM사)에 의해서 제공되는 z/아키텍처에 기초할 수 있다. z/아키텍처는 여기서 전체가 참조로 포함되는 IBM사가 2012년 8월 발표한 IBM 간행물 번호 SA22-7832-09에서 "z/아키텍처 연산의 원리들"이라는 제목의 IBM 발표물에 설명되어 있다. 한 예에서, z/아키텍처에 기초한 컴퓨팅 환경은 미국 뉴욕주 아몬크에 소재하는 인터내셔널 비지네스 머신즈 코포레이션(IBM사)에 의해서 제공되는, e서버 z시리즈(an eServer zSeries)를 포함한다. 컴퓨팅 환경은, 예를 들어, 하나 또는 그 이상의 코어들을 갖는 하나 또는 그 이상의 파티션들(예를 들어, 논리 파티션들)을 갖는 프로세서 복합체(a processor complex)와 하나 또는 그 이상의 하이퍼바이저들을 포함할 수 있는데, 이에 관해서는 후술한다.

[0019] [0019] 도 1a는 멀티스레딩(MT)을 지원하는 컴퓨팅 환경의 한 예로서 컴퓨터 시스템(100)을 도시한다. 도 1a의 예에서, 컴퓨터 시스템(100)은 복수의 프로세서 코어들(102), 입력/출력(I/O) 서브시스템(104), 및 시스템 메모리(160)을 포함한다. I/O 서브시스템(104)는 종래 기술에서 알려진 I/O 디바이스들에 대해 액세스를 제공한다. 프로세서 코어들(102)는, 또한 여기서 단순히 "코어들"로도 지칭되며, 지원 엘리먼트들을 갖는 처리 회로를 포

함할 수 있다. 도 1a의 예에서, 다섯 개의 코어들(102)는 코어1(110), 코어2(120), 코어3(130), 코어4(140), 및 코어5(150)으로 묘사된다; 그러나, 코어들(102)의 수는 이보다 더 많거나 더 적을 수도 있다. MT 퍼실리티(103)은 코어들(102)의 각각의 하드웨어 컴포넌트일 수 있다. 이 예에서, 코어들(102)의 각각은 4개의 스레드들까지 지원할 수 있다. 예를 들어, 코어1(110)은 스레드들(111, 112, 113, 및 114)를 지원할 수 있고, 코어2(120)는 스레드들(121, 122, 123, 및 124)를 지원할 수 있으며, 코어3(130)은 스레드들(131, 132, 133, 및 134)를 지원할 수 있고, 코어4(140)은 스레드들(141, 142, 143, 및 144)를 지원할 수 있으며, 코어5(150)은 스레드들(151, 152, 153, 및 154)를 지원할 수 있다. 여기서 주목할 것은 코어들(102)의 각각의 4개의 스레드들은 어떠한 순간에도 모두가 한꺼번에 동작될 수 있는 것은 아니라는 것이다. 예를 들어, 코어3(130)에서, 스레드들(131) 및 (132)는 동작될 수 있고, 한편 스레드들 (133) 및 (134)는 동작되는 것이 허용될 수 있다(음영으로 묘사됨).

[0020] [0020] 도 1a는 또한 컴퓨터 시스템(100)의 시스템 메모리(160)을 묘사하고, 이 시스템 메모리(160)은 논리 파티션(logical partition) 1 (LPAR1)(170), LPAR2(180), 및 LPAR3(190)에 할당된다. LPAR들(170, 180, 190)은 가상 컴퓨팅 시스템들(virtualized computing systems)(또한 구성들로도 알려짐)을 나타내며, Linux™ 또는 IBM® z/OS™, z/VM과 같은 운영 체제, 또는 zTPF 운영 체제가 이 논리 파티션들에서 실행될 수 있다. 도 1a는 또한 LPAR들(170, 180, 190)에 대한 코어들(102)의 할당을 도시한다. 이 예시에서, 코어1(110) 및 코어2(120)는 LPAR1(170)에 의한 사용을 위해 전속된다. 코어3(130)은 LPAR2(180)에 의한 사용을 위해 전속되고, 그리고 코어5(150)는 LPAR3(190)에 의한 사용을 위해 전속된다. 코어4(140)는 LPAR2(180) 및 LPAR3(190) 사이에서 공유될 수 있지만, 도 1a에서는 LPAR2(180)에 할당된 것으로서 도시되어 있다. LPAR3(190)은 파티션에 의해서 채용되는 두 개의 다른 타입들의 코어들(102)의 예를 보여주는데, 여기서 코어4(140)는 다수의 스레드들이 동작되는 것을 허용하지만, 코어5(150)는 이 예에서 다수의 스레드들이 동작되는 것을 허용하지 않는다. 도 1a의 예에서, LPAR1(170)은 OS(171) 및 프로그램들(172, 173, 174, 및 175)를 위해 처리 자원들을 제공한다. LPAR2(180)은 OS(181) 및 프로그램들(182, 183, 및 184)를 위해 처리 자원들을 제공한다. LPAR4(190)은 OS(191) 및 프로그램들(192, 및 193)을 위해 처리 자원들을 제공한다.

[0021] [0021] LPAR에서 실행하는 운영 체제의 컨트롤 하에서, 프로그램들은 코어의 스레드들 상에서 실행된다. 한 예시적 실시 예에서, 하나의 개별 스레드(an individual thread)는 한번에 오직 하나의 프로그램을 실행한다; 그러나, 재-실행(re-entrant)이 되도록 설계된 프로그램은 다수의 스레드들 또는 코어들에서 동시에 실행될 수 있다. 예를 들어, LPAR1(170)의 OS(171)의 프로그램(172)은 코어1(110)의 스레드들(111 및 113)에서 그리고 코어2(120)의 스레드들(121 및 124)에서 실행되고 있을 수 있다. OS의 컨트롤 하에서, 다른 프로그램들이 동일 또는 다른 스레드들에 파견될(dispatched) 수 있는데, 파견 규칙들 및 서비스 품질 합의서들에 따라(subject to dispatching rules and quality-of-service agreements) 그렇게 할 수 있다.

[0022] [0022] 시스템 메모리(160)에는 또한 다양한 수준들의 펌웨어가 상주하는데, 이들은, 예를 들어, 밀리코드(162) 및 LPAR 하이퍼바이저(163)을 포함한다. 밀리코드(162)는 하위-수준 시스템 기능들(lower-level system functions)을 지원하기 위해 펌웨어로서 구현될 수 있다. LPAR 하이퍼바이저(163)는, 예를 들어, IBM 프로세서-자원/시스템 매니저(the IBM Processor-Resource / System Manager™ (PR/SM™))와 같은 라이센스된 내부 코드(licensed internal code)일 수 있다. LPAR 하이퍼바이저(163)은 LPAR들(170, 180, 190)을 설립할 수 있고 코어들(102)에 파견하는 것(dispatching)을 관리할 수 있다. MT 퍼실리티(103)가 컴퓨터 시스템(100)에 설치될 때, 밀리코드(162) 및 LPAR 하이퍼바이저(163)은 또한 각각 MT 퍼실리티 지원 코드(164 및 165)를 포함한다. MT 퍼실리티 지원 코드(164 및 165)는, MT를 지원하기 위한 논리(logic)가 밀리코드(162), LPAR 하이퍼바이저(163), 및 코어들(102) 사이에서 배포될 때, MT 퍼실리티(103)의 일부로 고려될 수 있다. 비록 도시되지는 않았지만, OS들(171, 181, 191)의 각각은 또한 MT 퍼실리티 지원 코드를 포함할 수 있는데, 이는 그들 각각의 LPAR들(170, 180, 190)에서 MT를 동작 가능하게 하고 활용하기 위해서이다(enable and exploit).

[0023] [0023] 도 1b의 컴퓨팅 환경에서 코어4(140)이 LPAR2(180) 대신에 LPAR3(190)에 할당되었다는 점을 제외하고, 도 1b는 도 1a와 동일한 컴퓨터 시스템(100)을 도시한다. 또한 주목할 것은 스레드들(143 및 144)은 동작 가능 상태(operational)가 아닌, 도 1a와는 달리, 도 1b에서는, LPAR3(190)이 코어4(140)에 파견될 때 4개의 스레드들(141-144) 모두가 현재 동작 가능 상태에 있다는 점이다. 코어(102) 상에 LPAR을 파견하는 것과 파견을 철수하는 것(the dispatching and undispatching)은 동적으로 이루어지고, 다른 시간들에서 다른 LPAR들(도시하지 않음)이 동일 코어들(102)에서 동작하고 있을 수도 있다.

[0024] [0024] 이제 도 2를 살펴보면, 도 1a 및 1b의 코어들(102) 중의 하나와 같은, 처리 코어를 구현하기 위한 처리 회로(200)의 블록도가 한 실시 예에 따라 일반적으로 도시되어 있다. 처리 회로(200)은 MT 환경에서 하나 또는

그 이상의 스레드들을 동시에 지원할 수 있는 처리 회로의 예이다. 도 2에서 도시한 처리 회로(200)은 시스템 컨트롤러 인터페이스 유닛(a system controller interface unit) (202)를 포함하는데, 이는 처리 회로(200)을 다른 프로세서들 및 주변 장치들에 결합시킬 수 있다. 시스템 컨트롤러 인터페이스 유닛(202)는 또한, 데이터 값들을 읽고 저장하는 D캐시(a Dcache)(204), 프로그램 명령들을 읽는 I캐시(an Icache)(208), 및 캐시 인터페이스 유닛(a cache interface unit)(206)을 외부 메모리, 프로세서, 및 기타 주변 장치들에 연결할 수 있다.

[0025] [0025] I캐시(208)은 명령 페치 유닛(an instruction fetch unit (IFU))(210)과 함께 명령 스트림들의 로딩(loading)을 제공할 수 있고, 이 명령 페치 유닛(210)은 명령들을 프리페치(pre-fetch)하며 추측 로딩 및 분기 예측 능력들(speculative loading and branch prediction capabilities)을 포함할 수 있다. 페치된 명령들은 명령 처리 데이터로 디코드하기 위한 명령 디코드 유닛(an instruction decode unit (IDU))(212)에 제공될 수 있다.

[0026] [0026] IDU(212)는 명령들을 발행 유닛(an issue unit)(214)에 제공할 수 있고, 이 발행 유닛(214)는 명령들을 다양한 실행 유닛들, 예를 들어, 일반 연산들을 실행하는 하나 또는 그 이상의 고정 소수점 유닛들(fixed point units(FXU))(216) 및 부동 소수점 연산들을 실행하는 하나 또는 그 이상의 부동 소수점 유닛들(floating point units (FPU))(218)로 발행하는 것(issusing)을 컨트롤 할 수 있다. FPU들(218)은 이진 부동 소수점 유닛(BFU)(220), 십진 부동 소수점 유닛(DFU)(222), 또는 기타 부동 소수점 유닛을 포함할 수 있다. 발행 유닛(214)는 또한 하나 또는 그 이상의 로드/스토어 유닛들(load/store units (LSU))(228)에 결합될 수 있는데, 하나 또는 그 이상의 LSU 파이프라인들을 통해서 결합될 수 있다. 다수의 LSU 파이프라인들은 로드들 및 저장들 그리고 분기들을 위한 어드레스 생성을 수행하기 위한 실행 유닛들로서 취급된다. LSU(228) 및 IFU(210) 모두는 오퍼랜드 및 명령 어드레스들을 위한 버퍼된 변환들(buffered translations)을 제공하기 위해 변환-룩어사이드-버퍼(TLB)(230)을 이용할 수 있다.

[0027] [0027] FXU(216) 및 FPU(218)은 다양한 자원들, 예를 들어 범용 레지스터들(GPR)(224) 및 부동 소수점 레지스터들(FPR)(226)에 결합된다. GPR(224) 및 FPR(226)은 LSU(228)에 의해서 D캐시(204)로부터 로드되고 저장된, 데이터 값들을 위해 데이터 값 스토리지(data value storage)를 제공한다.

[0028] [0028] 처리 회로(200)은 또한 시스템 타임-베이스 생성 및 진단 액션들(system time-base generation and diagnostic actions)을 지원하기 위해 카운터들 및/또는 타이머들(250)을 포함할 수 있다. 예를 들어, 카운터들 및/또는 타이머들(250)은 다양한 진단 및 측정 퍼실리티들 뿐만 아니라 타임-오브-데이(TOD)를 지원하기 위해 사용될 수 있다.

[0029] [0029] 이제 도 3을 살펴보면, 도 3에서, 제2-수준의 하이퍼바이저(300)이 컴퓨터 시스템(100)의 LPAR2(180)에서 실행되고 있는 것을 제외하고, 도 1a와 유사한 컴퓨팅 환경이 도시되어 있다. 제2-수준의 하이퍼바이저(300), 예를 들어, IBM z/VM 운영 체제는 MT 지원 코드(301)을 포함하고, 이는 LPAR(제1-수준) 하이퍼바이저(163)에 의해서 제공된 MT 지원 코드(165)와 유사하다. 제2-수준의 하이퍼바이저(300)은 게스트 운영 체제들(311, 321, 및 331)이 각각 동작하는 복수의 가상 머신들(310, 320, 및 330)(이들을 또한 구성들이라 한다)에 대해 지원을 제공한다. 게스트 운영 체제들(311, 321, 및 331)은, 예를 들어, Linux™ 또는 IBM® z/OS™, z/VM, 또는 z/TPF OS를 포함하거나, 또는 IBM 대화 감시 시스템(the IBM conversational monitor system (CMS))과 같은 게스트 개발 환경을 포함할 수 있다. 각각의 게스트 OS들(311, 321, 및 331)은 멀티스레딩을 동작 가능하게 할 수도 또는 안 할 수도 있는데, 어느 경우든지 제2-수준의 하이퍼바이저(300)은 자신이 동작하는 LPAR2(180)에 이용 가능한 물리적 처리 자원들(코어들(130, 140) 및 스레드들(131-134, 141-144))을 이용하여 게스트 OS들(311, 321, 331) 및 관련 프로그램들(312, 313, 322, 323, 332, 및 333)을 파견하는 것에 대하여 책임이 있을 수 있다. 다양한 가상 머신들(310, 320, 330)의 프로그램들(312, 313, 322, 323, 332, 333)은 각각의 게스트 OS들(311,321, 및 331)에 이용 가능한 스레드들(131-134, 141-144)에서 실행할 수 있다. 만일 제2-수준의 하이퍼바이저(300)가 멀티스레딩을 이용한다면 게스트 OS들(311, 321, 및 331)은 MT로부터 투명하게 혜택을 입을 수 있으므로 그들은 MT 지원 코드를 포함할 필요가 없다.

[0030] [0030] 이제 도 4를 살펴보면, 한 실시 예에 따라 구현될 수 있는 컴퓨팅 환경에서 하이퍼바이저 컨텍스트 유지(hypervisor context retention)의 예가 도시되어 있다. 도 4의 예에서, 다수의 지원 구조들이 도 1a 및 1b의 LPAR 하이퍼바이저(163) 내에 묘사되어 있다. 예를 들어, 구조들(410)은 도 1a의 LPAR1(170)을 지원할 수 있고, 도 1a에서 도시한 물리적 스레드들(111, 112, 113, 114, 121, 122, 123, 124)에서 현재 실행하는 논리 스레드들(411, 412, 413, 414, 421, 422, 423, 424)을 위해 아키텍처된 레지스터 컨텍스트(architected register context)(즉, 스레드 컨텍스트)를 저장하는 상태 설명들(state descriptions(SD)) 및 위성 블록들(satellite

blocks)을 포함한다. 이들 논리 스레드들이 파견되어 있는 동안, 물리적 스레드들은 스레드들의 현재 아키텍처된 레지스터 컨텍스트를 보유한다. 그들이 더 이상 파견되지 않을 때 아키텍처된 레지스터 컨텍스트는 상태 설명들(SD) 및 위성 블록들 내에 유지될 것이다. 구조들(430)은 도 1a의 LPAR2(180)을 지원할 수 있고, 도 1a에서 도시한 물리적 스레드들(131, 132, 141, 142)에서 현재 실행하는 논리 스레드들(431, 432, 413, 441, 442)을 위해 아키텍처된 레지스터 컨텍스트를 저장하는 상태 설명들 및 위성 블록들을 포함한다. 구조들(450)은 도 1a의 LPAR3(190)을 지원할 수 있고, 도 1a에서 도시한 물리적 스레드들(151)에서 현재 실행하는 논리 스레드들(451)을 위해 아키텍처된 레지스터 컨텍스트를 저장하는 상태 설명들 및 위성 블록들을 포함한다. 구조들(450)은 또한 (음영(shading)으로 도시한) 물리적 프로세서에 현재 파견되지 않은 논리 스레드들(461, 462, 463, 및 464)을 위해 아키텍처된 레지스터 컨텍스트를 저장하는 상태 설명들 및 위성 블록들을 포함한다. 물리적 코어들에 파견되지 않은 LPAR들을 지원하는 기타 구조들도 또한 논리 스레드들(471, 472, 473, 및 474)을 위해 상태 설명들 및 위성 구조들을 포함하는 LPAR A(도1a에는 도시되지 않음)를 위한 구조들(470)과 같은, LPAR 하이퍼바이저에 의해서 유지될 수 있다. 추가의 구조의 예들도 논리 스레드(485)를 위해 비-파견된 LPAR C(도 1a에는 도시되지 않음)를 위한 구조들(484)뿐만 아니라, 논리 스레드들(481 및 482)를 위해 상태 설명들 및 위성 구조들을 포함하는 비-파견된 LPAR B(도 1a에는 도시되지 않음)를 지원하는 구조들(480)도 포함한다.

[0031] [0031] 다수의 구조들이 도 4의 예들에서 도시되었지만, 추가의 구조들이 멀티스레딩을 관리하기 위해 컴퓨터 시스템(100)에서 LPAR 하이퍼바이저(163) 및 기타에 의해서 지원될 수 있음을 이해할 수 있을 것이다. 예를 들어, 도 3의 가상 머신들(310, 320, 330)의 멀티스레딩을 지원하기 위한 구조들은 도3의 제2-수준의 하이퍼바이저(300)에 의해서 유지될 수 있다.

[0032] [0032] 이제 도 5를 살펴보면, 멀티스레딩의 동적 동작 가능을 위한 처리 흐름(500)이 한 실시 예에 따라 도시된다. 블록(502)에서, 1차 스레드는 단일 스레드(ST) 모드에서 실행한다. 블록(504)에서, 멀티스레딩(MT) 모드 세팅 명령이 ST 모드에서 페치된다. 블록(505)에서 집단적으로 묘사된 바와 같이 이 명령을 실행함에 있어서, MT 모드 세팅 명령들에 의해서 명시된 위치로부터 요청된 다수의 스레드들이 블록(506)에서 획득된다. 상기 위치는 MT 모드 세팅 명령을 발행할 때 파라미터 레지스터에 의해서 명시될 수 있다. MT 모드 세팅 명령은 요청된 다수의 스레드들과 관련된 세트-MT 지시(a set-MT order) 및 프로그램-명시된 최대 스레드-ID(PSMTID)를 포함하는 신호 프로세서(SIGP) 명령일 수 있다. SIGP 명령의 세트-MT 지시와 관련된 처리의 예는 도 7을 참조하여 더 설명할 것이다.

[0033] [0033] 처리(500)를 계속하여, 블록(508)에 이르면, 요청된 다수의 스레드들이 다수 스레드를 표시하는지에 관하여 결정이 수행된다. 예를 들어, 다수 스레드들은 1보다 큰 값에 의해서 표시될 수 있다. 영의 값이 단일 스레드를 표시하는 실시 예들에서, 1 또는 1보다 큰 값은 다수 스레드들을 표시할 수 있다. 다수의 요청된 스레드들이 다수 스레드들을 표시하지 않는다고 결정하는 것에 기초하여, 코어는 블록(510)에서 ST 모드에 남아있고, MT 모드 세팅 명령의 실행은 완료되며, 컨트롤은 블록(502)로 리턴한다. 다수의 요청된 스레드들이 다수 스레드들을 표시한다고 결정하는 것에 기초하여, MT 모드가 블록(512)에서 동작 가능하게 되고, MT 모드 세팅 명령의 실행은 완료된다. 블록(514)에서, 1차 스레드 및 하나 또는 그 이상의 2차 스레드들을 포함하는 다수 스레드들이 실행된다. 블록(516)에서, 만일 리셋 또는 비활성화가 존재한다면, 프로세스(500)은 루프를 통해 블록(514)로 되돌아간다; 그렇지 않으면, 블록(518)에서, MT 모드는 ST 모드로 복귀시키는 구성의 리셋 또는 비활성화에 기초하여 동작 가능하지 않게 된다. MT 모드를 동작 가능하지 않게 하는 것의 일부로서, 스레드들의 수(PSMTID)는 비-클리어링 리셋(a non-clearing reset)을 위해 유지되거나 또는 클리어링 리셋을 위해 0이된다(zeroed). 처리(500)은 블록(502)로 리턴한다.

[0034] [0034] CPU는 로드-노멀(a load-normal), 로드-위드-덤프(load-with-dump), 로드-클리어(load-clear), 또는 로드-클리어-리스트-디렉티드 키(load-clear-list-directed key)가 활성화될(activated) 때 로드 상태(a load state)로 진입할 수 있다. 만일 채널-명령 워드(CCW)-타입 초기-프로그램-로딩 연산(a channel-command word (CCW)-type initial-program-loading operation)이 성공적으로 완료되면, CPU는 로드 상태에서 연산 상태(the operating state)로 변경된다.

[0035] [0035] 파괴되는 정보의 양을 최소화하면서 장비-확인 표시들(equipment-check indications) 및 CPU 상태의 모든 최종 예측 불가능성(any resultant unpredictability in the CPU state)을 클리어하기 위해 CPU 리셋(reset)이 사용될 수 있다. 특히, CPU 상태가 분석 또는 연산의 재개를 위해 보존되어야 할 때 확인 조건들(check conditions)을 클리어하기 위해 사용될 수 있다. 만일 CPU 리셋이 로드-노멀 또는 로드-위드-덤프 키의 활성화에 의해서 발생되었다면, (a) 그것은 아키텍처 모드를 디폴트 모드로 세트시킬 수 있고, 그리고 (b) 만일 멀티스레딩 퍼실리티가 설치되어 있고 동작 가능하게 되어 있다면, 멀티스레딩은 동작 가능하지 않게 된다. CPU

리셋이 디폴트 모드를 세트할 때, 그것은 PSW가 복원될 수 있도록 현재의 PSW를 세이브할(save) 수 있다.

[0036] [0036] 초기 CPU 리셋은 현재 PSW, CPU 타이머, 클럭 비교기, 및 기타 레지스터들(예를 들어, 브레이킹-이벤트-어드레스(breaking-event-address), 캡처드-PSW(captured-PSW), 컨트롤(control), 부동-소수점-컨트롤(floating-point-control), 프리픽스(prefix), 및 TOD 프로그램 가능(TOD programmable) 레지스터들)의 초기화를 포함하여 CPU 리셋의 기능들을 제공한다. 초기 CPU 리셋은 만일 그것이 로드-노멀 또는 로드-위드-덤프 키의 활성화에 의해서 일어난 것이라면 아키텍처 모드를 디폴트 모드로 세트시킬 수 있다. 초기 CPU 리셋이 로드-노멀 또는 로드-위드-덤프 키의 활성화에 의해서 발생되었을 때 만일 멀티스레딩이 동작 가능하게 되어 있다면, 초기-CPU-리셋 기능들이 코어의 가장 낮은-번호의 CPU(the lowest-numbered CPU of a core)를 위해 수행될 수 있고, 그리고 CPU 리셋은 코어 내의 모든 다른 CPU들을 위해서 수행된다. 클리어링 리셋은 초기 CPU 리셋과 서브시스템 리셋이 수행되게 하고, 그리고 추가로 모든 스토리지 위치들 및 구성 내 모든 CPU들의 레지스터들을 클리어 또는 초기화하며, 다만 TOD 클럭은 예외로 한다. 클리어하는 것은 외부 스토리지에는 영향을 주지 않는데, 예를 들어 어드레스 할 수 없는 페이지들(unaddressable pages)의 컨텐츠를 보유하기 위해 컨트롤 프로그램에 의해서 사용되는 직접-액세스 스토리지 장치들에는 영향을 주지 않는다.

[0037] [0037] CPU 파워-온 리셋(a CPU power-on reset)은 초기 CPU 리셋이 수행되게 하여 일반 레지스터들, 액세스 레지스터들, 컨트롤 레지스터들, 및 부동-소수점 레지스터들의 컨텐츠를 0들/유효 확인-블록 코드(a valid checking -block code)를 갖는 디폴트 값들로 클리어 시킨다. 상태들을 클리어하는 것 또는 초기화하는 것이 값들을 0이 되게 할 필요는 없고 클리어된 상태에서 0이 아닌 값들(non-zero values)로 디폴트되게 할 수 있음을 이해하여야 한다. 만일 CPU 파워-온 리셋이 구성을 설정한다면, 그것은 아키텍처 모드를 디폴트 모드로 세트할 수 있다; 그렇지 않으면, 그것은 아키텍처 모드를 구성 내에 이미 존재하는 CPU들의 모드로 세트할 수 있다. CPU 리셋, 초기 CPU 리셋, 서브시스템 리셋, 및 클리어 리셋은 수동으로 개시될 수도 있다.

[0038] [0038] 예시적 실시 예들에서, 각각의 CPU는, 자신의 CPU 어드레스라 불리 우는, 할당된 번호를 가지고 있다. CPU 어드레스는 구성 내의 하나의 CPU를 고유하게 식별한다. CPU는 SIGP 명령의 CPU-어드레스 필드 내에 이 어드레스를 명시함으로써 지정된다. 고장 경보, 긴급 신호, 또는 외부 호출 신호를 전송하는 CPU(CPU signaling a malfunction alert, emergency signal, or external call)는 인터럽션(interruption)으로 CPU-어드레스 필드 내에 이 어드레스를 저장함으로써 식별될 수 있다. CPU 어드레스는 구성-정의 프로세스(configuration-definition process)에 의해서 할당되고, 통상적으로 재구성 변경들의 결과(a result of reconfiguration changes)로써 변경되지는 않는다. 프로그램은 CPU 어드레스 저장 명령(a store CPU address instruction)을 사용함으로써 CPU의 어드레스를 결정할 수 있다. CPU 어드레스 저장 명령은 또한 CPU 어드레스를 식별하기 위해서도 사용될 수 있고, CPU 어드레스에 의해서 CPU가 멀티프로세싱 구성에서 식별된다.

[0039] [0039] 멀티스레딩이 동작 가능하게 되어 있을 때, CPU 어드레스는, 코어 내의 CPU의 식별정보와 연결된(concatenated), 코어 식별 정보(코어 ID)를 포함할 수 있다. 코어 내의 CPU 식별정보는 스레드 식별정보(스레드ID, 또는 TID)이다. 구성 내에서, 모든 코어들은 동일 번호를 CPU들에 제공한다; 그러나, 모델 및 CPU 타입에 따라서, 코어 내 일부 CPU들은 동작 가능한 상태가 아닐 수 있다(not be operational).

[0040] [0040] 신호 프로세서 세트 멀티스레딩 지시(the signal processor set multithreading order)에 의해서 사용된 파라미터 레지스터의 PSMTID에 기초하여, 고정된 수의 비트들이 스레드 식별정보를 나타낸다. 이 비트들의 수는 TID 폭이라 한다.

[0041] [0041] 코어 ID는 멀티스레딩이 동작 가능하게 되기 전에 CPU 어드레스의 최우측 비트들(the rightmost bits)로부터 형성될 수 있다. 코어 ID는 TID-폭 비트들에 의해서 시프트되고(shifted), 그 결과 멀티스레딩이 이용 가능하게 된 후 CPU 어드레스의 최좌측 비트들(the leftmost bits)이 생성된다. 스레드 ID는 동일 TID-폭의 비트 수를 가지고, 멀티스레딩이 동작 가능하게 된 후 CPU 어드레스의 최우측 비트들을 차지한다. 스레드 ID들은 인접 범위의 수들(a contiguous range of numbers)에 할당될 수 있다. 표 1은 PSMTID, TID 폭 및 코어 식별 정보와 스레드 식별정보를 포함하는 CPU-어드레스 비트들의 예시적 관계를 도시한다.

표 1

[0042] 어드레스 비트 맵핑의 예

PSMTID	TID 폭	CPU 어드레스 비트들	
		코어 ID	스레드 ID
0	0	0-15	-

1	1	0-14	15
2-3	2	0-13	14-15
4-7	3	0-12	13-15
8-15	4	0-11	12-15
16-31	5	0-10	11-15

[0043] [0042] 한 실시 예에 따라 어드레스 확장(address expansion)이 CPU 어드레스 확장 프로세스(600A)의 한 예로서 도 6a에 도시되어 있다. 블록(602)에서, 1차 스레드(primary thread)는 코어 어드레스 값(604)을 다수의 CPU 어드레스 비트들로서 사용하여 ST 모드에서 어드레스 될 수 있다. 화살(606)은 ST 모드에서 MT 모드로 전환을 표시한다. 블록(608)에서, 1차 스레드 또는 하나 혹은 그 이상의 2차 스레드들이 확장된 어드레스 값(610)을 사용하여 MT 모드에서 어드레스 될 수 있다. 확장된 어드레스 값(610)은 시프트된 코어 어드레스 값(612)로 시프트되고 스레드 어드레스 값(614)와 연결된 코어 어드레스 값(604)를 포함한다. 시프트된 어드레스 값(612)는 코어 식별자(코어 ID)이고, 스레드 어드레스 값(614)는 스레드 식별자(TID)이다. 시프트된 코어 어드레스 값(612)는 요청된 최대 스레드 식별자, 예를 들어 PSMTID에 기초한 양에 의해서 시프트될 수 있다. 스레드 어드레스 값(614) 내의 다수의 TID 비트들은 위의 표 1에서 도시한 바와 같이 PSMTID에 기초하여 결정될 수 있다. 스레드 어드레스 값(614)는 확장된 어드레스 값(610)을 형성하기 위해 시프트된 코어 어드레스 값(612)의 하위 비트들(low order bits)에 연결될 수 있다. 스레드 어드레스 값(614)가 모두 0이면, 이는 1차 스레드를 지정하고, 0보다 큰 값이면, 2차 스레드들을 식별하고 어드레스한다.

[0044] [0043] MT 모드와 ST 모드 사이에 전환이 있을 때, 코어 어드레스 값(604)(ST 모드) 또는 확장된 어드레스 값(610)(MT 모드)은 각각의 ST 모드 또는 MT 모드에서 CPU어드레스로서 사용하기 위해 선택된다. 코어 어드레스 값(604)는 ST 모드에서 사용되는 표준-형식 어드레스(a standard-format address)의 예이고, 코어는 MT 모드를 동작 가능하지 않게 하는 것에 기초하여 MT 모드로부터 ST 모드로 복귀한다. 예시적 실시 예에서, 오직 1차 스레드(즉, 2차 스레드들이 아닌)만이 MT 모드를 동작 가능하지 않게 하는 것에 기초하여 액세스하는 것이 가능하다. 도 6b는 한 실시 예에 따라 CPU 어드레스 수축 프로세스(a CPU address contraction process)(600B)의 예를 도시한다. 도 6b의 화살(616)은 블록(608)의 MT 모드로부터 블록(602)의 ST 모드로 되돌아가는 전환을 도시한다. MT 모드로부터 ST 모드로의 복귀는 시프트된 코어 어드레스 값(612)로부터 CPU 어드레스로서 코어 어드레스 값(604)(코어 ID)를 포함하는 표준-형식 어드레스를 형성하기 위해 확장된 어드레스 값(610)을 우측으로 시프트하는 것과 스레드 어드레스 값(614)를 제거하는 것을 포함할 수 있다.

[0045] [0044] 리셋 기능이 멀티스레딩을 동작 가능하지 않게 할 때, (a) 스레드-ID 영(zero)을 갖는 CPU(들)의 CPU 어드레스(들)은 동작 가능 동안 사용되는 동일 TID-폭 수의 비트들에 의해 우측으로 시프트되고, (b) 상기 어드레스의 좌측 상의 TID-폭 수의 비트들 내에 0들이 삽입되며, 그리고 (c) 상기 CPU 어드레스는 자신의 오리지널 비-멀티스레딩 형식(즉, 표준-형식 어드레스)으로 복귀한다. 멀티스레딩이 동작 가능하게 될 때 0이 아닌 스레드 ID들을 갖는 코어 내 모든 CPU들은 멀티스레딩이 동작 가능하지 않게 될 때는 더 이상 동작 가능한 상태에 있지 않다(no longer operatonal).

[0046] [0045] 멀티스레딩이 동작 가능하게 되어 있지 않을 때, CPU 어드레스는 구성- 정의 프로세스에 의해서 할당된 값에서 변경이 없다. 이 경우에, 스레드 식별 정보는 존재하지 않는다.

[0047] [0046] 다수의 신호 프로세서 지시들(a number of signal processor orders)은, 예를 들어, 시작(start), 중단(stop), 재시작(restart), 중단 및 상태 저장(stop and store status), 초기 CPU 리셋(initial CPU reset), CPU 리셋(CPU reset), 어드레스에 상태 저장(store status at address), 아키텍처를 세트(set architecture), 실행 상태 감지(sense running status), 멀티스레딩을 세트(set multithreading), 어드레스에 추가 상태 저장(store additional status at address), 및 이와 유사한 것을 포함하는 지시들을 CPU들에 제공할 수 있다. 초기 CPU 리셋 또는 CPU 리셋은 신호 프로세서 명령에 의해서 개시될 수 있고, 아키텍처 모드 또는 기타 CPU들에 영향을 주지 않으며, 멀티스레딩을 동작 가능하지 않게 하지도 않고, I/O가 리셋되게 하지도 않는다.

[0048] [0047] 세트 아키텍처 지시(a set architecture order)는 구성 내 모든 CPU들이 세트되어야 하는 아키텍처 모드를 명시한다. 아키텍처 차이들은 다른 어드레싱 모드들, 레지스터 정의들, 및 CPU들에 의해 지원되는 명령들을 포함할 수 있다. 아키텍처 모드의 변경에 따라, 레지스터들의 비트 필드들을 선택함(select bit fields of registers)은 디폴트 상태로 세트될 수 있고(예를 들어, 0으로 됨), 구성 내의 모든 CPU들의 액세스-레지스터-변환 룩어사이드 버퍼들(access-register-translation lookaside buffers (ALB들)) 및 변환 룩어사이드 버퍼들(TLB들)은 클리어되며, 그리고 직렬화 및 체크포인트-동기화 기능(a serialization and checkpoint-

synchronization function)이 구성 내의 모든 CPU들에 관해 수행될 수 있다.

[0049] [0048] 실행 상태 감지 지시(a sense running status order)는 어드레스된 CPU가 실행 중인지를 표시할 수 있다. ST 모드에서, 표시자(an indicator)는 실행 중/실행 중 아님 상태(a running/not running status)로서 리턴될 수 있다. MT 모드에서, 표시자가 어드레스된 CPU가 멤버인 코어의 임의의 CPU가 실행 중에 있는지, 또는 어드레스된 CPU가 멤버인 코어의 모든 CPU들이 실행 중에 있지 않는지를 식별하기 위해 사용될 수 있다.

[0050] [0049] 세트-MT 지시는 멀티스레딩 퍼실리티를 동작 가능하게 한다. 파라미터 레지스터들의 비트 위치들은 구성 내에 제공될 PSMTID를 포함할 수 있다. PSMTID는 각각의 코어 내에 어드레스 가능하게 될 CPU들의 수보다 1 적은 값으로 정의될 수 있다. 예를 들어, 지정된 비트 위치들에서 3의 값은 4개의 스레드들 중 최대가 제공될 것이라는 것을 표시한다. SIGP 명령의 CPU-어드레스 레지스터의 컨텐츠는 구성 내의 모든 CPU들이 어드레스될 것으로 고려됨에 따라 무시될 수 있다. 만일 수용되면, 세트-MT 지시는 SIGP 명령의 실행 동안 모든 CPU들에 의해서 완료된다. 도 7을 참조하면, SIGP 세트-MT 지시(702)를 위한 처리(700)가 도시되어 있다. 에러 표시가 제공될 수 있고 그리고 MT 모드의 동작가능은 SIGP 세트-MT 지시(702)가 무효 지시(an invalid order), 부정확 상태(an incorrect state), 및 무효 파리미터(an invalid parameter) 중 하나 또는 그 이상과 함께 발행되었다고 결정하는 것에 기초하여 금지되며, 이에 관하여는 도 7의 처리(700)를 참조하여 더 상세하게 설명할 것이다.

[0051] [0050] 만일 멀티스레딩 퍼실리티가 블록(704)에서 설치되지 않았다거나 또는 CPU가 유효 아키텍처 모드(708)에서 동작 가능하게 되어 있지 않았다면, 세트-MT 지시는 수용되지 않고, 무효 지시 표시가 블록들(706 또는 710)에서 각각 리턴될 수 있다. 만일 구성 내의 다른 CPU들이 블록(712)에서 중단된 또는 체크-중단 상태(the stopped or check-stop state)에 있지 않거나 또는 만일 구성이 블록(716)에서 멀티스레딩을 위해 이미 동작 가능하게 되어 있다면, 상기 세트-MT 지시는 수용되지 않고 부정확 상태 표시가 블록들(714 또는 718)에서 각각 리턴될 수 있다.

[0052] [0051] 만일 PSMTID가 블록(720)에서 무효이면, 상기 세트-MT 지시는 수용되지 않고 무효 파리미터 표시가 블록(722)에서 리턴될 수 있다. PSMTID가 블록(724)에서 0일 때, 구성은 멀티스레딩을 위해 동작 가능하게 되지 않고, ST 모드에 남아있으며, 그리고 블록(728)에서 조건 코드(CC0)로서 임의의 상태를 제공한다. 한 예시적 실시예에서, PSMTID가 유효이고 0이 아닐 때, 블록(726)에서, 구성은 멀티스레딩을 위해 동작 가능하게 되고, 그 결과 CPU 어드레스 확장이 생성되고, 구성 내 모든 CPU들의 ALB들 및 TLB들은 그들의 컨텐츠가 클리어되며, 그리고 직렬화 및 체크포인트-동기화 기능이 구성 내 모든 CPU들에 관해 수행된다. 상태가 블록(728)에서 조건 코드에 제공될 수 있다. 성공적으로 완료되면, 상기 세트-MT 지시를 실행하는 CPU 외 모든 CPU들은 중단된 또는 체크-중단 상태에 남아있게 된다. 그러나, 만일 CPU가 멀티스레딩이 동작 가능하게 되기 전에 체크-중단 상태에 있었다면, 동일 코어 내 0이 아닌 스레드 ID들을 갖는 CPU들이 중단된 또는 체크-중단 상태에 배치되는지는 예측 불가능일 수 있다.

[0053] [0052] 스레드 컨텍스트(a thread context)는 또한 아키텍처된 레지스터 컨텍스트라 할 수 있다. 멀티스레딩이 동작 가능하게 되기 전 각각의 CPU의 아키텍처된 레지스터 컨텍스트(다시 말하면, PSW, CPU 타이머, 클럭 비교기, 일반 레지스터들, 부동-소수점 레지스터들 및 부동-소수점 컨트롤 레지스터, 벡터 레지스터들, 컨트롤 레지스터들, 액세스 레지스터들, 프리픽스 레지스터들, 및 TOD-프로그램 가능 레지스터, 등의 컨텐츠)는 멀티스레딩이 동작 가능하게 된 후 각각의 코어의 TID 0을 갖는 CPU의 아키텍처된 레지스터 컨텍스트가 된다. 유사하게, MT-동작 가능하게 된 구성의 각 코어의 TID 0을 갖는 CPU의 아키텍처된 레지스터 컨텍스트는 멀티스레딩이 로드-노멀 또는 로드-위드-덤프 키의 활성화의 결과로서 동작 가능하지 않게 될 때 각각의 CPU의 아키텍처된 레지스터 컨텍스트가 된다.

[0054] [0053] 0이 아닌 스레드 식별 정보를 갖는 모든 CPU들의 아키텍처된 레지스터 컨텍스트는 멀티스레딩 퍼실리티가 로드-노멀 또는 로드-위드-덤프 키 연산의 활성화의 결과로서 동작 가능하지 않게 될 때 유지될 수 있다. 만일 멀티스레딩 퍼실리티가 개입하는 클리어 리셋(an intervening clear reset) 없이 후속으로 재-동작 가능하게 된다면, 0이 아닌 스레드 식별 정보를 갖는 모든 CPU들의 아키텍처된 레지스터 컨텍스트는 복원된다.

[0055] [0054] 멀티스레딩이 로드-노멀 또는 로드-위드-덤프 키의 활성화에 의해서 동작 가능하지 않게 된 후 재-동작 가능하게 될 때, 만일 파리미터 레지스터의 비트들에서 PSMTID의 값이 이전의 동작가능(the preceding enablement)에서 사용된 값과 다르다면, 0이 아닌 스레드 ID들을 갖는 모든 CPU들의 아키텍처된 레지스터 컨텍스트는 예측 불가능하게 될 수 있다.

[0056] [0055] 시스템 정보 저장 명령(a store system information instruction)은 구성의 컴포넌트 또는 컴포넌트들

에 관한 정보를 시스템 정보 블록(SYSIB)에 저장하기 위해 사용될 수 있다. SYSIB는 MT 설치된 필드(an MT installed field), MT 일반 필드(an MT general field), 전체 CPU/코어 카운트(a total CPU/core count), 구성된 CPU/코어 카운트(a configured CPU/core count), 대기 CPU/코어 카운트(a standby CPU/core count), 유보된 CPU/코어 카운트(a reserved CPU/core count), 및 기타 필드들을 포함할 수 있다. MT 설치된 필드는 멀티스레딩 퍼실리티가 설치 되었는지를 표시할 수 있고, 또한 제1 코어 타입, 예를 들어, 특별 코어 타입을 위해 가장 높게 지원된 TID를 표시할 수도 있다. MT 일반 필드는 제2 코어 타입, 예를 들어 일반 코어 타입을 위해 가장 높게 지원된 TID를 표시할 수도 있다. MT 일반 필드에서 가장 높게 지원된 TID는 MT 설치된 필드에서 가장 높게 지원된 TID보다 적거나 또는 같도록 제한될 수 있다. 구성된 상태이든지, 대기 상태이든지, 또는 유보된 상태이든지, 구성 내 일반 CPU들을 포함하는 전체 CPU/코어 카운트는 일반 CPU들 또는 코어들 수의 합계를 표시할 수 있다. 구성된 CPU/코어 카운트는 구성된 상태에 있는, 즉 구성이 되어 프로그램들을 실행할 준비가 된 상태에 있는 일반 CPU들을 포함하는 일반 CPU들 또는 코어들의 수를 표시할 수 있다. 대기 CPU/코어 카운트는 대기 상태에 있는, 즉 구성된 상태에 배치될 때까지 프로그램들을 실행하기 위해 사용되도록 이용 가능하지 않은 상태에 있는 일반 CPU들을 포함하는 일반 CPU들 또는 코어들의 수를 표시한다. 유보된 CPU/코어 카운트는 유보된 상태에 있는, 즉 프로그램들을 실행하기 위해 사용되도록 이용 가능하지도 않고 구성된 상태에 배치될 수도 없는 상태에 있는 일반 CPU들을 포함하는 일반 CPU들 또는 코어들의 수를 표시한다.

[0057] [0056] 도 8은 한 실시 예에 따라 멀티스레딩 능력 정보를 저장하는 것의 예를 도시한다. 코어(800A)의 스레드 1과 같은, 스레드에서 실행 중인 프로그램은 시스템 정보 저장 명령(a STORE SYSTEM INFORMATION (STSI) instruction) (830)을 LPAR과 같은 구성(850)의 메모리(801)로부터 페치할 수 있다. STSI 명령의 실행은 시스템 예에서, SYSIB(802)는 구성(850)이 멀티스레딩을 지원하는지를 표시하는 MT 설치된 식별자 정보 블록(SYSIB)(802)을 저장(832) 하는 결과를 가져올 수 있다. 도 8의 예에서, SYSIB(802)는 구성(850)이 멀티스레딩을 지원하는지를 표시하는 MT 설치된 식별자(804)를 포함한다. SYSIB(802)는 또한 특별 코어들을 위한 코어(806) 당 최대 TID 및 일반 코어들(808)을 위한 최대 TID로서 제공될 수 있는 코어(800A/800B)의 가장 높게 지원된 스레드의 최대 스레드 식별자를 포함한다. SYSIB(802)는 또한 현재 프로그램-명시된 최대 스레드 식별자(PSMTID)(809)를 포함할 수 있다. 현재 PSMTID(809)는 프로그램에 의해서 구성(850)에 동작 가능하게 될 때 멀티스레딩 모드를 반영한다. 현재 PSMTID(809)는 만일 STSI 명령(830)이 베이직-머신 수준(a basic-machine level)에서 실행된다면 정의되지 않을 수 있다.

[0058] [0057] 코어(800B)의 스레드 2와 같은, 스레드에서 실행 중인 프로그램은 또한 서비스 호출 명령(a SERVICE CALL (SERVC) instruction)(834)를 구성(850)의 메모리(801)로부터 페치할 수 있고, 이 명령은 리드-시스템-컨트롤-프로그램-정보 커맨드(a read-system-control-program-information (리드-SCP-정보, 또는 RSCPI) command)를 명시한다. 상기 RSCPI 커맨드의 실행은 서비스-호출 컨트롤 블록(SCCB)(810)이 메모리(801)에 저장되도록 할 수 있다. 한 예시적 실시 예에서, RSCPI 커맨드의 실행에 의해서 저장된 SCCB(810)은 SYSIB(802)에서 이용 가능하지 않을 수 있는 유사한 그리고 추가의 정보를 제공한다. 도 8의 예에서, SCCB(810)은 코어(800B)가 멀티스레딩을 지원하는지를 표시하는 MT 설치된 식별자(812)를 포함한다. SCCB(810)은 또한 특별 코어들을 위한 코어(814) 당 최대 TID 및 일반 코어들(816)을 위한 최대 TID로서 제공될 수 있는 코어(800B)의 가장 높게 지원된 스레드의 최대 스레드 식별자를 포함한다. SCCB(810)의 값들(812-816)은 SYSIB(802)에서 액세스 가능할 수 있는 값들(804-808)과 같다(equivalent). 추가로, SCCB(810)은 코어(800B)의 가장 높게 지원된 스레드의 최종-세트 프로그램-명시된 최대 스레드 식별자(a last-set program-specified maximum thread identifier)를 포함할 수 있고, 이를 또한 최종-세트 프로그램-명시된 최대 스레드 식별자(PSMTID)(818)라 한다. SCCB(810)은 또한 PSMTID 지원된 마스크(820)으로서 세트-MT 지시 상에서 수용 가능한 PSMTID 값들의 마스크를 포함할 수 있다. PSMTID 지원된 마스크(820)은 코어(814) 당 최대 TID에 의해서 정의된 수보다 더 적은 수가 필요할 때 지원된 CPU들/스레드들을 식별하기 위해 사용될 수 있다.

[0059] [0058] 코어들(800A 및 800B)은 이 예에서 설명되지 않은 다른 특징들을 포함한다는 것을 이해해야 한다. 더 나아가, SYSIB(802) 및 SCCB(810)은 도 8의 예에서 설명한 것 이상의 추가적인 값들을 포함할 수 있다.

[0060] [0059] 도 9는 한 실시 예에 따라 멀티스레딩 능력을 결정하기 위한 처리 흐름(900)을 도시한다. 블록(902)에서, 코어는 멀티스레딩 능력 정보 검색 명령(a retrieve multithreading capability information (RMTCI) instruction)을 실행하는데, 이 명령은, 예를 들어, SERVC 명령 또는 STSI 명령 중 어느 하나일 수 있다. 블록(904)에서, 구성의 멀티스레딩 능력을 식별하는 스레드 식별 정보가 획득된다. 블록(906)에서, 획득된 스레드 식별 정보가 저장된다. 블록(908)에서, 구성이 이전에 획득된 스레드 식별 정보에 기초하여 멀티스레딩을 동작 가능하게 되게 하였는지가 결정된다.

[0061] [0060] 전술한 바와 같이, SERVC 명령은 메모리 내의 응답 블록(예를 들어, 도 8의 SCCB(810))에 스레드 식별 정보를 저장하도록 구성되고, STSI 명령은 메모리 내의 SYSIB(예를 들어, 도 8의 SYSIB(802))에 스레드 식별 정보를 저장하도록 구성된다. 획득된 스레드 식별 정보는 코어가 멀티스레딩을 지원하는지를 표시하는 MT 설치된 식별자(an MT installed identifier)(예를 들어, 도 8의 MT 설치된 식별자(804 또는 812))를 포함할 수 있다. 획득된 스레드 식별 정보는 또한 코어의 가장 높게 지원된 스레드의 최대 스레드 식별자(예를 들어, 도 8의 최대 TID 값들(806, 808, 814, 또는 816))를 포함할 수 있다. 획득된 스레드 식별 정보는 현재 프로그램-명시된 최대 스레드 식별자(a current program-specified maximum thread identifier)(예를 들어, 도 8의 현재 PSMTID(809)) 및 최종-세트 프로그램-명시된 최대 스레드 식별자(a last-set program-specified maximum thread identifier)(예를 들어, 도 8의 PSMTID(818))를 포함할 수 있다. 응답 블록은 개별적으로 지원되는 특정 스레드 식별자들을 표시하는 비트들의 마스크(예를 들어, 도 8의 PSMTID지원된 마스크)를 포함할 수 있다. 구성이 이전에 MT를 동작 가능하게 했다는 결정은 최종-세트 프로그램-명시된 최대 스레드 식별자 내의 0이 아닌 값에 기초할 수 있다(예를 들어, 최종-세트 PSMTID > 0). 한 예시적 실시 예에서, 구성은 복수의 코어 타입들을 지원한다.

[0062] [0061] 예시적 실시 예들에서, 레지스터들 및 값들, 예를 들어 프로그램 카운터 값들 – 이들은 레지스터들 내에 포함되거나 또는 별도로 관리될 수 있음 – 은 스레드 컨텍스트로서 캡처된다(captured). 어드레스 확장이 MT 모드에서 발생될 때, 추가적인 스레드 컨텍스트가 액세스 가능하게 된다. 도 6을 참조하여 전술한 바와 같이, CPU 어드레스는 구성 내 각각의 코어에 대해서 형성된다. CPU 어드레스는 CPU 어드레스 저장 명령(the store CPU address instruction)에 의해서 점검될 수 있고(inspected), 그것은 다른 구조들에서도 나타나며, 그리고 그것은 다양한 SIGP 지시들에서도 사용된다. MT가 동작 가능하게 되지 않았을 때, 이 어드레싱 방식은 변경되지 않는다. MT가 동작 가능하게 되었을 때, CPU 어드레스는 확장 프로세스로 진행한다. 전술한 바와 같이, CPU 어드레스의 비-MT-동작 가능하게 된 부분(the non-MT-enabled portion)은 TID를 수용하기 위해서 충분한 비트들이 좌측으로 시프트될 수 있다. 예를 들어, 만일 운영 체제가 PSMTID 값 1을 갖는 SIGP 세트-MT 지시를 발행하였다면, CPU 어드레스는 1 비트만큼 좌로 시프트될 것이다; 만일 PSMTID가 2 또는 3이었다면, CPU 어드레스는 2 비트만큼 좌로 시프트될 것이고, 만일 PSMTID가 4-7이라면, CPU 어드레스는 3 비트만큼 좌로 시프트될 것이다, 등등.

[0063] [0062] 멀티스레딩이 후속적으로 동작 가능하지 않게 될 때(로드-노멀 연산에 의해서 발생된 클리어 리셋 또는 CPU 리셋의 결과로서), CPU 어드레스 수축이 일어난다. MT-동작 가능하게 된 CPU 어드레스는 MT를 동작 가능하게 한 SIGP 세트-MT 지시에 사용된 동일 PSMTID 비트 수만큼 우측으로 시프트될 수 있고, 그리고 상기 어드레스의 스레드-ID 부분은 사라진다. MT 모드 동안 액세스 가능한 스레드 컨텍스트는, 도 10에서 묘사된 예와 같은, 하나 또는 그 이상의 위치들에 상주할 수 있다. 도 10의 예에서, 구성(1000)은 코어(1002)를 포함하고 그리고 다른 코어들(도시하지 않음)도 포함할 수 있다. 메모리(1006)은 구성(1000)의 일부로서 구성 메모리(1005)와 구성(1000)으로부터 분리된 호스트/펌웨어 메모리(1007)을 포함할 수 있다. 호스트/펌웨어 메모리(1007)은 호스트에 의해서 유지되는 상태-설명 블록(a state-description block)(1008) – 스레드(예를 들어, 도 10의 스레드 n)를 위해 스레드 컨텍스트(1010)을 저장할 수 있음 – 을 포함할 수 있다. 위성 블록(a satellite block)(1012)는 호스트/펌웨어 메모리(1007)의 일부분으로서 메모리(1006) 내의 상태-설명 블록(the state-description block)(1008)에 고정될(anchored) 수 있고, 위성 블록(1012)은 스레드 컨텍스트(1014)를 스레드 컨텍스트(1010)에 대한 대안(alternative)으로서 또는 스레드 컨텍스트(1010)과 결합하여 포함할 수 있다. 각각의 스레드는 대응 상태-설명 블록(1008)과 선택적으로 위성 블록(1012)를 가질 수 있고, 스레드 컨텍스트(1010) 또는 스레드 컨텍스트(1014)는 저장될 수 있다. 또 다른 대안으로서, 하드웨어 컨텍스트 레지스터들(1016)이 스레드 컨텍스트(1018)을, 예를 들어 코어(1002) 내에 저장하기 위해 사용될 수 있다. 스레드 컨텍스트(1010, 1014, 및 1018)의 예들은 스토리지 선택들로서 함께 또는 별도로 사용될 수 있다. 실시 예들에서 대체 스토리지 선택들이 채용될 수 있다. 스레드 컨텍스트가 어디에서 유지되든지 상관없이, 어드레스 수축에 따라, 스레드 컨텍스트는 더 이상 직접적으로 액세스할 수 있지 않지만, 덤프 프로그램(a dump program)에 의한 액세스를 위해 보존될 수 있다.

[0064] [0063] MT가 동작 가능하지 않게 될 때, CPU-어드레스-수축 프로세스는 코어의 스레드들 1-n을 더 이상 어드레스 가능하지 않게 한다; 비슷하게, 아키텍처된 레지스터들을 포함하는 스레드 컨텍스트는 더 이상 프로그램에 대해 보여질 수 있는(visible) 상태가 아니다. 만일 MT가 비-클리어링 로드 연산(a non-clearing load operation)으로부터 초래되는 CPU 리셋의 결과로서 동작 가능하지 않게 되었다면, 스레드들 1-n의 레지스터 컨텍스트는 유지된다; 이들 데이터는 만일 구성이 MT 모드로 리턴된다면 후속적으로 점검될 수도 있다. 각각의 게

스트 스레드를 위한 레지스터 컨텍스트는 도 10에서 묘사한 바와 같이 호스트에 의해서 스레드의 상태-설명 블록(the thread's state-description block)(1008)에 유지될 수 있다(또는, 벡터 레지스터들의 경우에서처럼, 상태 설명에 고정된 위성 블록(1012)에 유지될 수 있다).

[0065] [0064] MT가 동작 가능하지 않는 동안 스레드들 1-n의 컨텍스트의 유지(retention)는 OS 실패 다음에 덤프될 스레드들의 상태에 대한 진단 특징(a diagnostic feature)이다. OS 실패 다음에, 운영자는 OS 실패 타임에서(at the time of OS failure) 시스템의 메모리 및 스레드 컨텍스트를 캡처하기 위해 독립-덤프 프로그램(a stand-alone-dump (SADMP) program)을 실행하는 것을 선택할 수 있다. 그러나, SADMP 프로그램을 로드하는 것은 구성으로 하여금, ST 모드가 동작 가능하게 되고, 따라서 MT가 동작 가능하지 않게 되는, 디폴트 아키텍처 모드로 복귀하게 할 수 있다. 그러나, SADMP는 비-클리어링 로드 연산에 의해서 로드되기 때문에, 각 코어의 스레드들 1-n의 레지스터 컨텍스트는 유지된다. SADMP는 SERVC 리드-SCP-정보 커맨드의 응답 블록(a SERVC read-SCP-information command's response block)의 결과들을 조사함에 의해서 덤프되는 구성에서 MT가 동작 가능하게 되었는지를 결정할 수 있다. 이 수(This number)는 전과 동일 수준에서 MT를 재-동작 가능하게 하기 위해 후속적으로 SIGP 세트-MT 지시(the SIGP set-MT order)에 대한 입력으로서 사용될 수 있다.

[0066] [0065] 도 11은 한 실시 예에 따라 멀티스레딩 레지스터 보존의 한 예를 도시한다. 도 11의 컴퓨터 시스템(1100)과 같은, 시스템은 다수의 구성들(1102 및 1104)를 포함할 수 있다. 도 11의 예에서, 구성(1102)는 코어(1106) 및 코어(1108)을 포함하고, 구성(1104)는 코어(1110) 및 코어(1112)를 포함한다. 구성들(1102 및 1104)의 각각은 다른 타임들에서 ST 및 MT 모드들 간에 독립적으로 전환될 수 있다. 컴퓨터 시스템(1100)의 구성들(1102 및 1104)의 각각은 다른 수의 최대 스레드-ID들로 구성이 가능한데, 이는 구성들(1102 및 1104)의 각각에서 동시에 동작 가능하게 된 다른 수의 스레드들을 지원하기 위함이다. 도 11의 예에서, 코어들(1106 및 1108) 각각은 구성(1102)가 MT 모드(1114)에 있는 동안 최대 두 개의 스레드들을 지원하고, 반면에, 코어들(1110 및 1112) 각각은 구성(1104)가 MT 모드(1116)에 있는 동안 최대 네 개의 스레드들을 지원한다.

[0067] [0066] 구성(1102)에서 MT 모드(1114)가 동작 가능하게 되어 있는 동안, TID0 및 TID1 모두는, 스레드 컨텍스트(1115)의 별도의 인스턴스들(separate instances)과 같은, 별도의 스레드 컨텍스트들로서 액세스 가능하다. 타임(1118)에서, MT 모드(114)는 로드-노멀 연산 또는 구성(1102)에 대한 비-클리어링 리셋 – 이는 코어들(1106 및 1108) 모두를 ST 모드(1120)으로 전환함 – 에 의해서 동작 가능하지 않게 될 수 있다. 전술한 바와 같은 어드레스 수축으로 인하여, TID0 레지스터들은 ST 모드(1120)에서 액세스 가능하다; 그러나, MT 모드(1114)에서 액세스 가능하였던 TID1 레지스터들은 유지는 되지만 더 이상 액세스 가능하지는 않다. 예를 들어, TID1 레지스터들은 도 10의 스레드 컨텍스트(1010, 1014 또는 1018)로서 구현될 수 있고, 어드레스 확장으로 이용 가능하였던 어드레스는 ST 모드(1120)으로의 전환에 따른 어드레스 수축 후에는 더 이상 액세스 가능하지 않다.

[0068] [0067] 구성(1104)가 MT 모드(1116)를 동작 가능하게 되게하는 동안, TID0, TID1, TID2, 및 TID3 레지스터들은, 도 10의 스레드 컨텍스트(1010, 1014, 또는 1018)의 별도의 인스턴스들과 같은, 별도의 스레드 컨텍스트로서 액세스 가능하다. 이 예에서, TID0은 1차 스레드를 나타내고 TID1-TID3은 코어들(1110 및 1112)의 각각에 대해 별도로 유지되던 2차 스레드들을 나타낸다. 타임(1122)에서, MT 모드(1116)은 구성(1104)에 대한 클리어링 리셋 – 이것은 코어들(1110 및 1112) 모두를 ST 모드(1124)로 전환시킴 – 에 의해서 동작 가능하지 않게 될 수 있다. 타임(1122)에서의 클리어링 리셋은 TID0, TID1, TID2, 및 TID3의 모든 레지스터들을 클리어할 수 있다. 전술한 바와 같은 어드레스 수축으로 인하여, TID0 레지스터들은 ST 모드(1124)에서 액세스 가능하다; 그러나 MT 모드(1116)에서 액세스 가능하였던 TID1, TID2, 및 TID3 레지스터들은 클리어된 상태에서 유지는 되지만 더 이상 액세스 가능하지는 않다. 도 11에서 도시한 바와 같이, 연산들은 각각의 구성(1102 및 1104) 상에서, 다른 타임들(1118 및 1122)에서, 그리고 각각의 구성(1102 및 1104)에 로컬화된 효과들을 가지고 독립적으로 수행될 수 있다. 따라서, 구성(1102)는 구성(1104)가 MT 모드(1116)에 있는 동안 ST 모드(1120)에 있을 수 있으며, 그리고 ST/MT 모드들은 컴퓨터 시스템(1100)의 모든 구성들에 대해 정렬될 필요는 없다.

[0069] [0068] 도 12는 한 실시 예에 따라 멀티스레딩 레지스터 보존을 위한 처리 흐름(1200)을 도시한다. 블록(1202)에서, MT가 코어 내에서 동작 가능하지 않게 될 것이라고 MT 모드에서 코어가 결정하는 것에 기초하여, MT 모드로부터 ST 모드로의 전환이 수행된다. MT 모드의 1차 스레드는 ST 모드의 유일한(only) 스레드로서 유지될 수 있다. 2차 스레드들의 프로그램 액세스 가능 레지스터 값들과 프로그램 카운터 값들을 포함하는 하나 또는 그 이상의 스레드 컨텍스트들은 애플리케이션 프로그램들에는 액세스 가능하지 않게 되어 있다. 블록(1204)에서, 전환에 기초하여, 연산 타입(예를 들어, 클리어링 또는 비-클리어링)이 결정되어, 프로그램 액세스 가능 레지스터 값들을 클리어하는 것이 수행되거나 또는 프로그램 액세스 가능 레지스터 값들을 유지하는 것이 수행된다. 블록(1206)에서, 비-클리어링 연산에 기초하여, 프로그램 액세스 가능 레지스터들이 유지되는 것이 결정된다.

블록(1208)에서, 클리어링 연산에 기초하여, 프로그램 액세스 가능 레지스터들이 클리어되는 것이 결정된다.

[0070] [0069] 전술한 바와 같이, 스레드 컨텍스트의 프로그램 액세스 가능 레지스터 값들 및 프로그램 카운터 값들은 프로그램 범용 레지스터들, 부동-소수점 레지스터들, 컨트롤 레지스터들, 액세스 레지스터들, 프리픽스 레지스터들, 및 TOD 프로그램 가능 레지스터들을 포함할 수 있다. 컨트롤 레지스터들은 부동-소수점 컨트롤 레지스터, 런타임 계측 컨트롤들, CPU-측정 컨트롤들, 및 이와 유사한 것을 포함할 수 있다. 스레드 컨텍스트에 포함될 수 있는 레지스터들의 기타 예들은 프로그램-상태 워드(program-status word)(예를 들어, 프로그램 카운터/명령 어드레스, 조건 코드, 및 명령 시퀀싱을 제어하고 CPU 상태를 결정하기 위한 기타 정보를 포함하는), 벡터 레지스터들, CPU 타이머, 클럭 비교기, 브레이킹-이벤트 어드레스 레지스터, 및 기타 종래에 알려진 레지스터들을 포함한다. 전술한 바와 같이, PSMTID는 MT가 동작 가능하게 되도록 해준 성공적으로 실행된 최종 신호 프로세서 명령(a last successfully executed signal processor instruction)에 기초하여 세트된다. MT 모드로의 전환에 기초하여, 프로그램 액세스 가능 레지스터 값들은 대응 2차 스레드들이 재-동작 가능하게 된다는 것에 기초하여 애플리케이션 프로그램들에 액세스 가능하게 된다. 예를 들어, 도 11에서 ST 모드(1120)으로부터 MT 모드(1114)로 전환하면, TID1 레지스터들이 액세스 가능하게 되고, 그리고 TID1은 재-동작 가능하게 될 수 있다. 도 10의 스레드 컨텍스트(1010, 1014, 또는 1018)과 같은, 스레드 컨텍스트는: 상태-설명 블록(a state-description block), 메모리 내 상태-설명 블록에 고정된 위성 블록(a satellite block anchored to the state-description block in memory), 또는 컨텍스트 레지스터(a context register) 중 어느 하나에서 유지될 수 있다.

[0071] [0070] 1차 스레드 컨텍스트는 1차 스레드의 프로그램 액세스 가능 레지스터 값들 및 프로그램 카운터 값들, 예를 들어, 도 11의 구성(1104)를 위한 TID0 및 TID1 레지스터들을 포함할 수 있고, 1차 스레드 컨텍스트는 ST 모드(1124) 및 MT 모드(1116) 모두에서 애플리케이션 프로그램들에 대해 액세스 가능하다. 2차 스레드 컨텍스트는 2차 스레드의 프로그램 액세스 가능 레지스터 값들 및 프로그램 카운터 값들, 예를 들어, 도 11의 구성(1104)를 위한 TID1-TID3 및 TID1-TID3 레지스터들을 포함할 수 있다.

[0072] [0071] 도 13은 한 실시 예에 따라 멀티스레딩 레지스터 복원의 한 예를 도시한다. 도 13의 예는 단일 구성(1302)을 갖는 컴퓨터 시스템(1300)을 포함한다. 구성(1302)는 코어(1304), 코어(1306), 및 코어(1308)을 포함한다. 코어들(1304-1308)의 각각은 이 예에서 최대 4개의 스레드들(TID0, TID1, TID2, 및 TID3)을 포함한다. MT 모드(1310)에서, TID0-TID3의 모든 스레드 컨텍스트들이 코어들(1304-1308)에서 이용 가능하다. 타임(1312)에서, MT 모드(1310)은 로드-노멀 연산 또는 구성(1302)의 비-클리어링 리셋 – 이는 코어들(1304-1308)을 ST 모드(1314)로 전환시킴 – 에 의해서 동작 가능하지 않게 될 수 있다. ST 모드(1314)에서, TID0 레지스터들은 액세스 가능하고, TID1-TID3 레지스터들은 액세스 가능하지 않지만 코어들(1304-1308)의 각각을 위해 유지된다. 타임(1316)에서, MT는 재개된 MT 모드(1318)에 들어가기 위해서 SIGP 세트-MT 지시의 실행에 의해 재-동작 가능하게 될 수 있다. 재개된 MT 모드(1318)에서, 코어들(1304-1308)의 각각을 위한 TID1-TID3 레지스터들의 스레드 컨텍스트에 대한 액세스는 복원된다. 이것은, 분석을 위한 스레드 컨텍스트 정보를 세이브하기 위해, 독립형 덤프 프로그램(stand-alone dump program)(1320)과 같은, 덤프 프로그램에 의해, TID1-TID3 레지스터들을 포함하는, 모든 스레드의 레지스터들의 점검(inspection)을 동작 가능하게 한다.

[0073] [0072] 도 14는, 운영 체제의 실패 다음에 스레드들의 아키텍처된 레지스터 컨텍스트를 캡처하기 위해서, 도 13의 독립형 덤프 프로그램(1320)과 같은, 독립형 덤핑(SADMP) 프로그램에 의해서 채용될 수 있는, 한 실시 예에 따른 멀티스레딩 레지스터 복원을 위한 처리 흐름(1400)을 도시한다. 블록(1405)에서, SADMP 프로그램은 비-클리어링 로드 연산(예를 들어, 로드 노멀 또는 로드-위드-덤프)을 통해 로드 된다. 비-클리어링 로드 연산은 묵시적으로 구성을 ST 모드, 예를 들어 도 13의 구성(1302)를 위한 ST 모드(1314)로 복귀시킨다. SADMP 프로그램은 그 다음, STSI 또는 SERVC 명령을 사용함에 의해, 블록(1410)에서 MT 퍼실리티가 구성 내에서 이용 가능한지 질의(query) 할 수 있다. 만일 MT가 설치되어 있다면, SADMP 프로그램은 블록(1415)에서 구성을 위해 세트되는 최종-세트 프로그램-명시된 최대 스레드 식별정보(the last-set program-specified maximum thread identification (PSMTID))에 대해 질의한다. 만일 MT가 구성을 위해 이전에 결코 세트된 적이 없다면, 상기 최종-세트 PSMTID 값은 0이 될 것이다. SADMP 프로그램은 그 다음 상기 최종-세트 PSMTID가 어떤 값이었든지 간에 (비록 그 값이 0이었다 하더라도) 블록(1420)에서 멀티스레딩을 재-동작 가능하게 하기 위해 상기 명령을 실행할 수 있다. 만일 블록(1410)에서의 질의가 MT는 설치되어 있지 않음을 알려준다면, 블록(1415)에서 상기 최종-세트 PSMTID 값에 대해 질의하거나 또는 블록(1420)에서 MT를 재-동작 가능하게 하려는 시도는 하지 않는다.

[0074] [0073] SADMP 프로그램은 메모리 내 미리 정의된 위치에 자신의 아키텍처된 레지스터 컨텍스트를 세이브하기 위해 구성 내 각각의 다른 CPU(스레드)에 신호를 전송하려 한다. 만일 MT가 SADMP를 로드하기 이전에 미리 동작 가능하게 되어 있지 않다면, CPU 어드레스들은 노멀이고, 비-확장된 형식이다. 만일 MT가 미리 동작 가능하게

되었다면, CPU 어드레스들은 코어 ID와 스레드 ID를 포함하는 확장된 형식이다. SADMP는 블록(1425)에서 0인 CPU 에드레스(N)으로 시작하고, 블록(1430)에서 그 CPU 어드레스가 SADMP가 현재 실행 중인 CPU를 나타내는지를 결정한다. 만일 그렇다고 결정한다면, 그 CPU/스레드는 스킵되고(skipped), 블록(1450)에서 다음 값으로 증가된다(incremented). 만일 N이 현재 CPU 어드레스와 다르다면, 그 CPU/스레드는 블록(1435)에서, 예를 들어, SIGP 스토어-상태-엣-어드레스 지시(a SIGP store-status-at-address order) 또는 SIGP-스탑-엔드-스토어-상태 지시(a SIGP stop-and-store-status order)의 실행에 의해서, 메모리에 자신의 아키텍처된 레지스터 컨텍스트를 저장하도록 신호가 전송된다(signaled). 만일 구성이 벡터 퍼실리티를 포함한다면, SIGP 스토어-추가의-상태-엣-어드레스 지시(a SIGP store-additional-status-at-address order)가 또한 그 CPU/스레드의 벡터 레지스터들의 컨텐츠를 저장하도록 실행될 수 있다. 블록(1435)의 신호가 블록(1440)에서 성공적이었는지에 관해 결정이 내려진다. 만일 성공적이었다면, SADMP 프로그램은 블록(1445)에서 테이프 또는 디스크 상의 덤프 파일에 CPU/스레드의 레지스터 컨텍스트를 세이브할 수 있고, 그리고 처리는 계속되어 블록(1450)에서 N을 증가시킨다. 만일 블록(1435)의 신호가 블록(1440)에 의해서 결정된 바와 같이 성공적이지 않다면(예를 들어, 만일 스레드가 동작 가능 상태(operational)가 아니라면), 그것은 스킵되고(skipped), 블록(1450)에서 N을 증가시킴으로써 처리는 계속된다. 신호 전송(N)에서 사용된 CPU 어드레스의 값은 블록(1450)에서 증가되고, 블록(1455)에서 N이 이제 구성을 위한 가장 높은-가능한 CPU 어드레스(the highest-possible CPU address)보다 더 큰지에 관하여 결정이 내려진다. 만일 N이 구성을 위한 가장 높은-가능한 CPU 어드레스보다 더 크지 않다면, 블록(1430)에서 N이 SADMP 프로그램이 실행 중인 현재 CPU/스레드를 나타내는지를 결정함으로써 처리는 계속된다. 만일 N이 구성을 위한 가장 높은-가능한 CPU 어드레스보다 더 크다면, 블록(1460)에서 아키텍처된 레지스터 컨텍스트 복원과 덤핑(dumping)이 완료된다.

[0075] [0074] 도 14에서 구성의 한 코어에 관하여 기술하였지만, 도 14의 처리 흐름 (1400)은 다수의 코어들을 포함하는 구성의 모든 코어들에 걸친 최대 CPU 어드레스를 통해 실행하도록 확장될 수 있음을 이해하여야 한다. MT를 지원하지 않는 OS 또는 MT-인지(aware)는 하지만 MT를 이용하지 않는 프로그램을 위한 덤핑을 지원하기 위해 추가적인 수용들(additional accomodations)이 구성들 내 이루어질 수 있다. 예를 들어, MT-인지-독립형 덤프 프로그램(an MT-aware stand-alone dump program)이 구성으로부터 임의의 2차 스레드들을 덤프 하려는 것을 방지하기 위해 구성에서 MT를 지원하지 않는 OS를 로드하기 전에 클리어링 리셋이 수행될 수 있다. 다른 예로서, MT-인지는 하지만 MT를 이용하지 않는 프로그램은, 구성을 위해 독립형 덤프 프로그램을 실행하기 전에 0의 대응 최대 스레드-ID를 갖는 세트-MT 지시(a set-MT order)를 발행할 수 있다.

[0076] [0075] 기술적 효과들 및 장점들에는 연산의 단일 스레드 모드 및 멀티스레딩 모드 모두를 지원하는 멀티스레딩 컴퓨터 시스템에서 어드레스 확장 및 수축이 포함된다.

[0077] [0076] 여기서 설명한 시스템은 MT 하드웨어를 이용하도록 OS에게 명시적으로 "참여(opt in)"할 것을 요구함으로써 하드웨어 변동성을 완화하기 위해 소프트웨어를 동작 가능하게 한다. OS가 실행 환경의 MT 특성을 이해할 때, OS는 코어-당 스레드 밀도(the per-core thread densities)(워크로드 파견 패턴(a workload dispatch pattern)이 주어질 때, 최상의 능력까지)를 명시적으로 관리할 능력을 갖는다. OS는 계산 자원들이 적게 이용될 때조차도 높은 스레드 밀도를 유지하기 위한 선택을 갖고 있고, 이에 의해서 다른 MT 구현들에서 보여지는 전체 계산 능력에서 변동성을 상당히 완화할 수 있다. 높은 스레드 밀도를 유지하는 직접적인 결과로서, 트랜잭션 응답 시간들 및 빌링 특징들이 더 일관될 수 있다(more consistent).

[0078] [0077] 실시 예들은 멀티스레딩 컴퓨터 시스템에서 어드레스 확장 및 수축을 위한 시스템, 방법, 및 컴퓨터 프로그램 제품을 포함한다. 한 실시 예에 따라 컴퓨터 시스템은 단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에 구성 가능한 코어를 갖는 구성을 포함한다. 상기 ST 모드는 1차 스레드를 어드레스하고, 상기 MT 모드는 1차 스레드 및 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스한다. 코어 어드레스 값을 사용하여 상기 ST 모드에서 1차 스레드를 액세스하고 상기 ST 모드로부터 상기 MT 모드로 전환하는 단계(switching)를 포함하는 방법을 수행하기 위해 상기 구성의 이용을 컨트롤하도록 멀티스레딩 퍼실리티(a multithreading facility)가 구성된다. 상기 1차 스레드 또는 상기 하나 또는 그 이상의 2차 스레드들 중 하나는 확장된 어드레스 값을 사용하여 상기 MT 모드에서 액세스되고, 상기 확장된 어드레스 값은 스레드 어드레스 값과 연결된 상기 코어 어드레스 값을 포함한다.

[0079] [0078] 다른 실시 예에 따라, 구성에서 어드레스 조정을 위한 컴퓨터 구현 방법이 제공된다. 상기 구성은 단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에 구성 가능한 코어를 포함하고, 상기 ST 모드는 1차 스레드를 어드레스하며, 상기 MT 모드는 1차 스레드 및 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스한다. 상기 1차 스레드는 코어 어드레스 값을 사용하여 상기 ST 모드에서 액세스된다. 상기 ST 모드로부터

상기 MT 모드로 전환하는 단계(switching)가 수행된다. 상기 1차 스레드 또는 상기 하나 또는 그 이상의 2차 스레드들 중 하나는 확장된 어드레스 값을 사용하여 상기 MT 모드에서 액세스된다. 상기 확장된 어드레스 값은 스레드 어드레스 값과 연결된 상기 코어 어드레스 값을 포함한다.

[0080] [0079] 또 다른 실시 예는 구성에서 어드레스 조정을 위한 컴퓨터 프로그램 제품을 포함한다. 상기 구성은 단일 스레드(ST) 모드와 멀티스레딩(MT) 모드 사이에서 구성 가능한 코어(a core configurable)를 포함하고, 상기 ST 모드는 1차 스레드(a primary thread)를 어드레스하며(address) 그리고 상기 MT 모드는 상기 1차 스레드와 상기 코어의 공유 자원들 상의 하나 또는 그 이상의 2차 스레드들을 어드레스한다. 상기 컴퓨터 프로그램 제품은 컴퓨터 판독가능 스토리지 매체를 포함하고, 상기 컴퓨터 판독가능 스토리지 매체는 신호가 아니며, 상기 컴퓨터 판독가능 스토리지 매체에는 프로그램 명령들이 구현되어 있다. 상기 프로그램 명령들은 처리 회로가 코어 어드레스 값을 사용하여 상기 ST 모드에서 1차 스레드를 액세스하고 상기 ST 모드로부터 상기 MT 모드로 전환하는 단계(switching)를 포함하는 방법을 수행하도록 상기 처리 회로에 의해서 판독가능하다. 상기 1차 스레드 또는 상기 하나 또는 그 이상의 2차 스레드들 중 하나는 확장된 어드레스 값을 사용하여 상기 MT 모드에서 액세스되고, 상기 확장된 어드레스 값은 스레드 어드레스 값과 연결된 상기 코어 어드레스 값을 포함한다.

[0081] [0080] 전술한 하나 또는 그 이상의 실시 예들에 더하여, 또는 대안으로서, 추가의 실시 예들을 포함할 수 있고, 여기서 상기 확장된 어드레스 값은 시프트된 코어 어드레스 값(a shifted core address value)을 포함하고, 상기 시프트된 코어 어드레스 값은 요청된 최대 스레드 식별자에 기초하는 양 만큼(by an amount based on a requested maximum thread identifier) 시프트된다.

[0082] [0081] 전술한 하나 또는 그 이상의 실시 예들에 더하여, 또는 대안으로서, 추가의 실시 예들을 포함할 수 있고, 여기서 상기 스레드 어드레스 값은 상기 확장된 어드레스 값을 형성하기 위해 상기 코어 어드레스 값의 하위 비트들(low order bits)에 연결된다(concatenated).

[0083] [0082] 전술한 하나 또는 그 이상의 실시 예들에 더하여, 또는 대안으로서, 추가의 실시 예들은 상기 MT 모드와 상기 ST 모드 사이에서 전환하는 단계를 포함할 수 있고, 상기 코어 어드레스 값 또는 상기 확장된 어드레스 값 중 하나는 상기 코어가 각각 ST 모드 또는 MT 모드에 있는 것에 기초하여 선택된다.

[0084] [0083] 전술한 하나 또는 그 이상의 실시 예들에 더하여, 또는 대안으로서, 추가의 실시 예들을 포함할 수 있고, 여기서 표준-형식 어드레스(a standard-format address)가 상기 ST 모드에서 사용되고, 상기 코어는 상기 MT 모드가 동작 가능하지 않게 되는 것(disabling)에 기초하여 상기 MT 모드로부터 상기 ST 모드로 복귀한다.

[0085] [0084] 전술한 하나 또는 그 이상의 실시 예들에 더하여, 또는 대안으로서, 추가의 실시 예들을 포함할 수 있고, 여기서 상기 MT 모드가 동작 가능하지 않게 되는 것(disabling)에 기초하여 오직 상기 1차 스레드만 액세스 가능하게 되고, 상기 하나 또는 그 이상의 2차 스레드들은 액세스 가능하지 않게 된다.

[0086] [0085] 전술한 하나 또는 그 이상의 실시 예들에 더하여, 또는 대안으로서, 추가의 실시 예들을 포함할 수 있고, 여기서 상기 MT 모드로부터 상기 ST 모드로의 복귀(reversion)는 상기 확장된 어드레스 값을 시프트하는 단계(shifting)와 상기 스레드 어드레스 값을 제거하는 단계(eliminating)를 더 포함한다.

[0087] [0086] 전술한 하나 또는 그 이상의 실시 예들에 더하여, 또는 대안으로서, 추가의 실시 예들을 포함할 수 있고, 여기서 상기 스레드 어드레스 값에서 다수의 스레드 식별자 비트들(a number of thread identifier bits)은 프로그램-명시된 최대 스레드 식별자(a program-specified maximum thread identifier)에 기초하여 결정된다.

[0088] [0087] 여기서 사용된 용어는 특정 실시 예들을 기술할 목적으로만 사용된 것이며 발명의 범위를 제한하려 의도된 것이 아니다. 여기서 사용된 바와 같이, 단수 형식들은, 달리 문맥 상 명백하게 표시하지 않는 한, 복수의 형식들도 또한 포함하도록 의도 되었다. “포함하다” 및/또는 “포함하는”이라는 용어들은, 이 명세서에서 사용되었을 때, 진술된 특징들, 정수들, 단계들, 연산들, 엘리먼트들, 및/또는 컴포넌트들의 존재를 명시하지만, 하나 혹은 그 이상의 다른 특징들, 정수들, 단계들, 연산들, 엘리먼트들, 컴포넌트들 및/또는 그들의 그룹들의 존재 혹은 추가를 배제하는 것이 아님을 이해하여야 한다.

[0089] [0088] 아래 청구항들 내의 대응 구조들, 재료들, 동작들, 및 모든 수단들 혹은 단계 플러스 기능 엘리먼트들의 균등물들은 명시적으로 청구된 다른 엘리먼트들과 조합하여 기능을 수행하기 위한 모든 구조, 재료, 혹은 동작을 포함하도록 의도된다. 본 발명의 상세한 설명은 예시와 설명의 목적으로 제공되었지만, 개시된 형식 내의 발명이 전부라거나 혹은 한정되도록 의도되지 않았다. 본 발명의 범위와 정신을 벗어남이 없이 많은 수정들과 변경들이 당업자들에게 명백하다. 실시 예는 본 발명의 원리들과 실용적 응용을 최선으로 설명하기 위해서, 그리

고 당업자들이 고려된 특정 사용에 적합한 다양한 수정들을 갖는 다양한 실시 예들에 대해 본 발명을 이해할 수 있도록 하기 위해서 선택되고 기술되었다.

[0090] [0089] 본 발명의 다양한 실시 예들에 관한 상세한 설명은 예시의 목적으로 제공되었지만, 개시된 실시 예들이 전부라거나 혹은 한정되도록 의도되지 않았다. 기술된 실시 예들의 범위와 정신을 벗어남이 없이 많은 수정들과 변경들이 당업자들에게 명백하다. 여기서 사용된 용어들은 실시 예들의 원리들, 실용적 응용 또는 시장에서 발견된 기술들에 관한 기술적 개선을 최선으로 설명하기 위해서, 또는 여기서 개시된 실시 예들을 당업자들이 이해할 수 있도록 하기 위해서 선택되고 기술되었다.

[0091] [0090] 이제 도 15를 참조하면, 컴퓨터 판독가능 스토리지 매체(1502) 및 프로그램 명령들(1504)를 포함하는 한 실시 예에 따른 컴퓨터 프로그램 제품(1500)이 일반적으로 도시되어 있다.

[0092] [0091] 본 발명은 시스템, 방법, 및/혹은 컴퓨터 프로그램 제품이 될 수 있다. 컴퓨터 프로그램 제품은 프로세서가 본 발명의 특징들을 수행하도록 그 위에 컴퓨터 판독가능 프로그램 명령들을 갖는 컴퓨터 판독가능 스토리지 매체들(또는 매체)을 포함할 수 있다.

[0093] [0092] 컴퓨터 판독가능 스토리지 매체는 명령 실행 디바이스에 의한 사용을 위해 명령들을 보유하고 저장할 수 있는 유형의 디바이스일 수 있다. 컴퓨터 판독가능 스토리지 매체는, 예를 들어, 전자 스토리지 디바이스, 자기 스토리지 디바이스, 광 스토리지 디바이스, 전자기 스토리지 디바이스, 반도체 스토리지 디바이스, 혹은 전술한 모든 이들의 적절한 조합이 될 수 있으며, 이에 한정되지 않는다. 컴퓨터 판독가능 스토리지 매체의 더 구체적인 예들은 다음을 포함할 수 있다: 휴대용 컴퓨터 디스켓, 하드 디스크, 랜덤 액세스 메모리(RAM), 판독-전용 메모리(ROM), 지울 수 있고 프로그램 가능한 판독-전용 메모리(EPROM 혹은 플래시 메모리), 정적 랜덤 액세스 메모리(SRAM), 휴대용 콤팩트 디스크 판독-전용 메모리(CD-ROM), 디지털 다용도 디스크(DVD), 메모리 스틱, 플로피 디스크, 홈에 명령들을 기록하는 펀치-카드들 또는 융기된 구조물들과 같은 기계적으로 인코딩된 디바이스, 및 이들의 모든 적합한 조합. 여기서 사용되는, 컴퓨터 판독가능 스토리지 매체는 라디오 전파들 또는 기타 자유롭게 전파되는 전자파들, 도파관 또는 기타 전송 매체를 통해서 전파되는 전자파들과 같은 전송 신호들 그 자체(예를 들어, 광-섬유 케이블을 통해 전송되는 광 펄스들), 또는 전선을 통해 전달되는 전기 신호들인 것으로 해석되지는 않는다.

[0094] [0093] 여기서 기술한 컴퓨터 판독가능 프로그램 명령들은 컴퓨터 판독가능 스토리지 매체로부터 각 컴퓨팅/처리 디바이스들로 다운로드될 수 있거나 또는, 예를 들어 인터넷, 근거리 통신 네트워크, 원거리 통신 네트워크 및/또는 무선 네트워크와 같은 네트워크를 통해 외부 컴퓨터 또는 외부 스토리지 디바이스에 다운로드될 수 있다. 상기 네트워크에는 구리 전송 케이블들, 광 전송 섬유들, 무선 전송, 라우터들, 방화벽들, 스위치들, 게이트웨이, 컴퓨터들 및/또는 엣지 서버들이 포함될 수 있다. 각 컴퓨팅/처리 디바이스 내 네트워크 어댑터 카드 또는 네트워크 인터페이스는 네트워크로부터 컴퓨터 판독가능 프로그램 명령들을 수신하여, 그 컴퓨터 판독가능 프로그램 명령들을 각 컴퓨팅/처리 디바이스 내의 컴퓨터 판독가능 스토리지 매체 내에 저장하기 위해 전송한다.

[0095] [0094] 본 발명의 연산들을 수행하기 위한 컴퓨터 판독가능 프로그램 명령들은 하나 이상의 프로그래밍 언어들의 조합으로 작성된, 어셈블러 명령들, 명령-세트-아키텍처(ISA) 명령들, 머신 명령들, 머신 종속 명령들, 마이크로코드, 펌웨어 명령들, 상태-세팅 데이터, 또는 소스 코드 또는 객체 코드가 될 수 있으며, 이들 언어들은 Smalltalk, C++ 혹은 이와 유사한 것과 같은 객체 지향 프로그래밍 언어와, "C" 프로그래밍 언어 혹은 이와 유사한 프로그래밍 언어들과 같은 종래의 절차적 프로그래밍 언어들을 포함한다. 컴퓨터 판독가능 프로그램 명령들은 전적으로 사용자의 컴퓨터 상에서, 부분적으로 사용자의 컴퓨터 상에서, 독립형 소프트웨어 패키지로서, 실행될 수 있고, 부분적으로 사용자 컴퓨터 상에서 그리고 부분적으로 원격 컴퓨터 상에서 또는 전적으로 원격 컴퓨터 혹은 서버 상에서 실행될 수 있다. 후자의 시나리오에서, 원격 컴퓨터는 사용자의 컴퓨터와 연결될 수 있는데, 이는 근거리 네트워크(LAN) 혹은 광역 네트워크(WAN)를 포함하는 모든 종류의 네트워크를 통해서 할 수 있거나, 또는 연결은, 예를 들어 인터넷 서비스 제공자를 사용하는 인터넷을 통해서, 외부 컴퓨터로 이루어질 수 있다. 일부 실시 예들에서, 예를 들어, 프로그램 가능 논리 회로, 필드-프로그램 가능 게이트 어레이(FPGA), 또는 프로그램 가능 논리 어레이들(PLA)을 포함하는, 전자 회로가 본 발명의 특징들을 수행하기 위해서, 컴퓨터 판독가능 프로그램 명령들을 수행할 수 있는데, 상기 전자회로를 맞춤형으로 구성(personalize)하도록 컴퓨터 판독가능 프로그램 명령들의 상태 정보를 이용함으로써 할 수 있다.

[0096] [0095] 본 발명의 여러 특징들이 본 발명의 실시 예들에 따른 방법들, 장치(시스템들) 및 컴퓨터 프로그램 제품들의 흐름도 예시들 및/또는 블록도들을 참조하여 기술되었다. 흐름도 예시들 및/또는 블록도들의 각 블록 및,

흐름도 예시들 및/또는 블록도들 내의 블록들의 조합들은 컴퓨터 판독가능 프로그램 명령들에 의해서 구현될 수 있음을 이해하여야 한다.

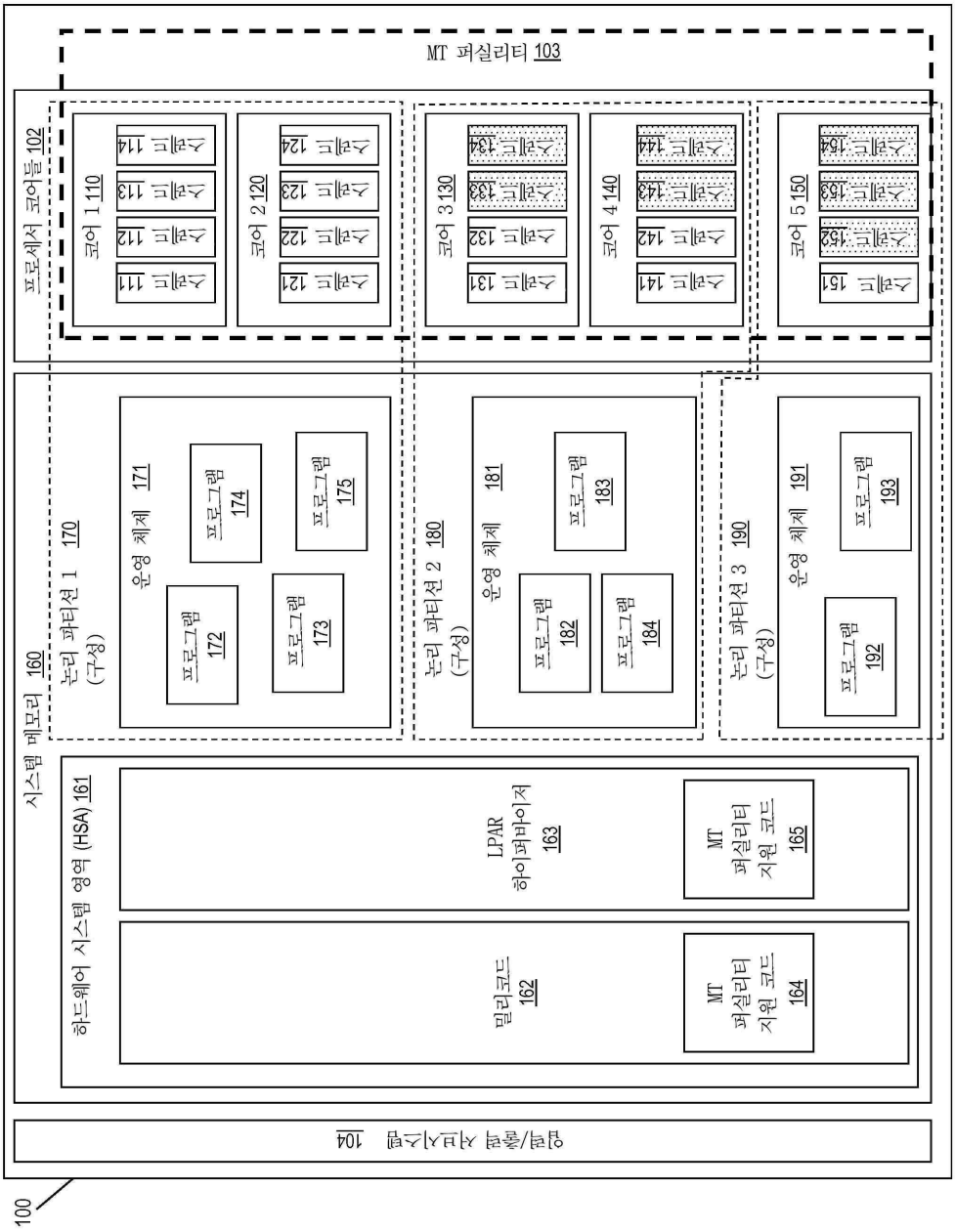
[0097] [0096] 이들 컴퓨터 판독가능 프로그램 명령들은 범용 컴퓨터, 특별 컴퓨터, 혹은 프로그램 가능 데이터 처리 장치의 프로세서에 제공되어, 컴퓨터 혹은 기타 프로그램 가능 데이터 처리 장치의 프로세서를 통해서 실행하는, 명령들이 흐름도 및/또는 블록도 블록 또는 블록들 내에 명시된 기능들/동작들을 구현하는 수단들을 생성하도록 머신을 생산할 수 있다. 이들 컴퓨터 판독가능 프로그램 명령들은 또한 컴퓨터 판독가능 매체 내에 저장될 수 있고, 상기 명령들은 컴퓨터, 프로그램 가능 데이터 처리 장치, 및/또는 기타 디바이스들이 특정 방식으로 기능을 수행하도록 지시할 수 있으며, 따라서 컴퓨터 판독가능 스토리지 매체 내에 저장된 명령들을 갖는 컴퓨터 판독가능 스토리지 매체가 흐름도 및/또는 블록도 블록 또는 블록들 내에 명시된 기능/동작의 특징들을 구현하는 명령들을 포함하는 제품을 포함한다.

[0098] [0097] 컴퓨터 판독가능 프로그램 명령들은 또한 컴퓨터, 기타 프로그램 가능 데이터 처리 장치, 혹은 기타 디바이스들에 로드되어 컴퓨터, 기타 프로그램 가능 데이터 처리 장치, 혹은 기타 디바이스들 상에서 실행하는 명령들이 흐름도 및/또는 블록도 블록 또는 블록들 내에 명시된 기능들/동작들을 구현하도록, 컴퓨터 구현된 프로세스를 생산하기 위해서 컴퓨터, 기타 프로그램 가능 데이터 처리 장치, 혹은 기타 디바이스들 상에서 일련의 동작 단계들이 수행되게 할 수 있다.

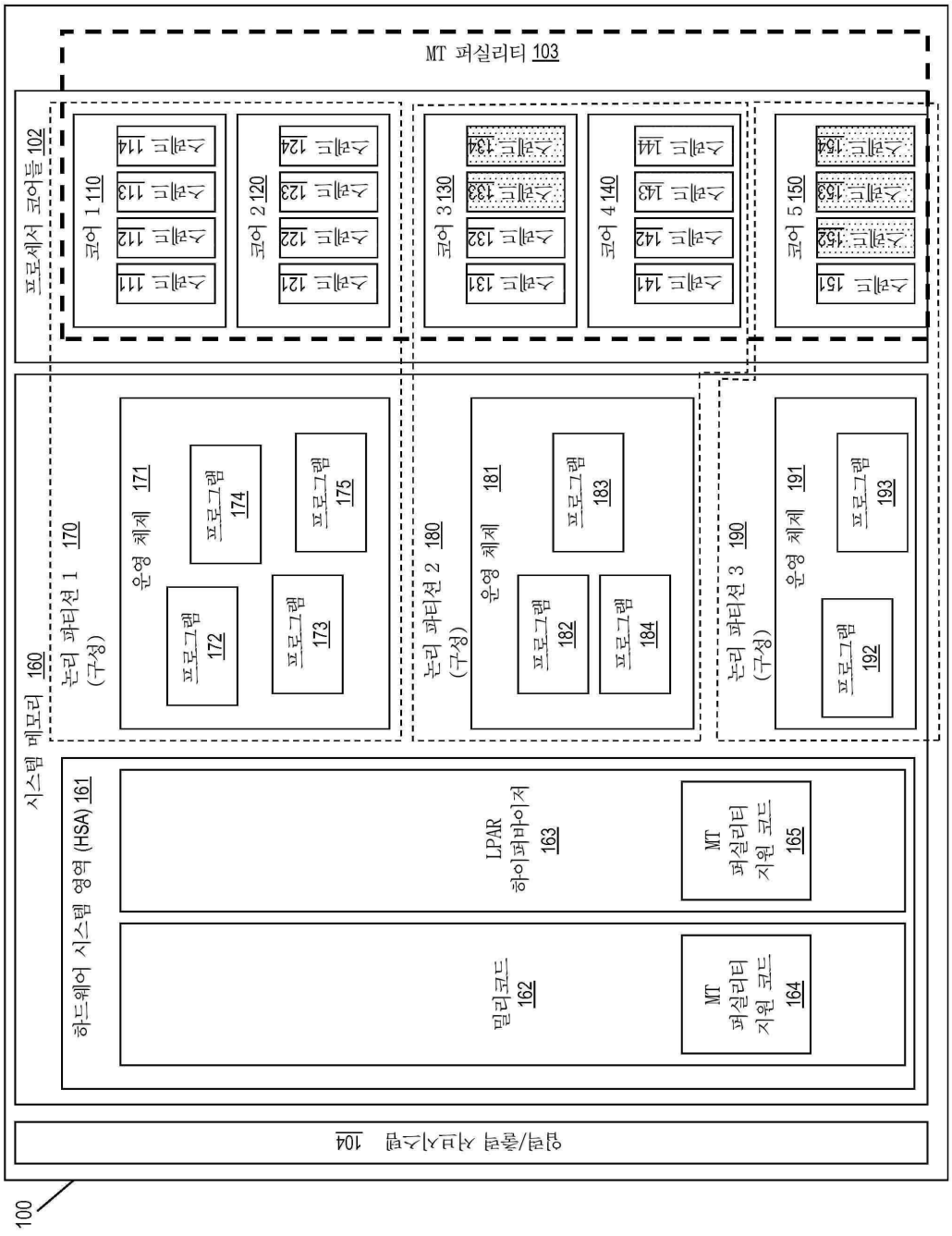
[0099] [0098] 도면들 내 흐름도 및 블록도들은 본 발명의 여러 실시 예들에 따른 시스템들, 방법들 및 컴퓨터 프로그램 제품들의 가능한 구현들의 아키텍처, 기능성, 및 연산을 예시한다. 이와 관련하여, 상기 흐름도 또는 블록도들 내 각 블록은 상기 명시된 논리 기능(들)을 구현하기 위한 하나 또는 그 이상의 실행 가능한 명령들을 포함한 모듈, 세그먼트 또는 명령들의 일부분을 나타낼 수 있다. 일부 다른 구현들에서, 상기 블록에 언급되는 기능들은 도면들에 언급된 순서와 다르게 일어날 수도 있다. 예를 들면, 연속으로 도시된 두 개의 블록들은, 실제로는, 사실상 동시에 실행될 수 있고, 또는 이 두 블록들은 때때로 관련된 기능성에 따라서는 역순으로 실행될 수도 있다. 블록도들 및/또는 흐름도 예시도의 각 블록, 및 블록도들 및/또는 흐름도 예시도 내 블록들의 조합들은 특수목적용 하드웨어 및 컴퓨터 명령들의 명시된 기능들 또는 동작들, 또는 이들의 조합들을 수행하는 특수목적용 하드웨어-기반 시스템들에 의해 구현될 수 있다는 것에 또한 유의한다.

도면

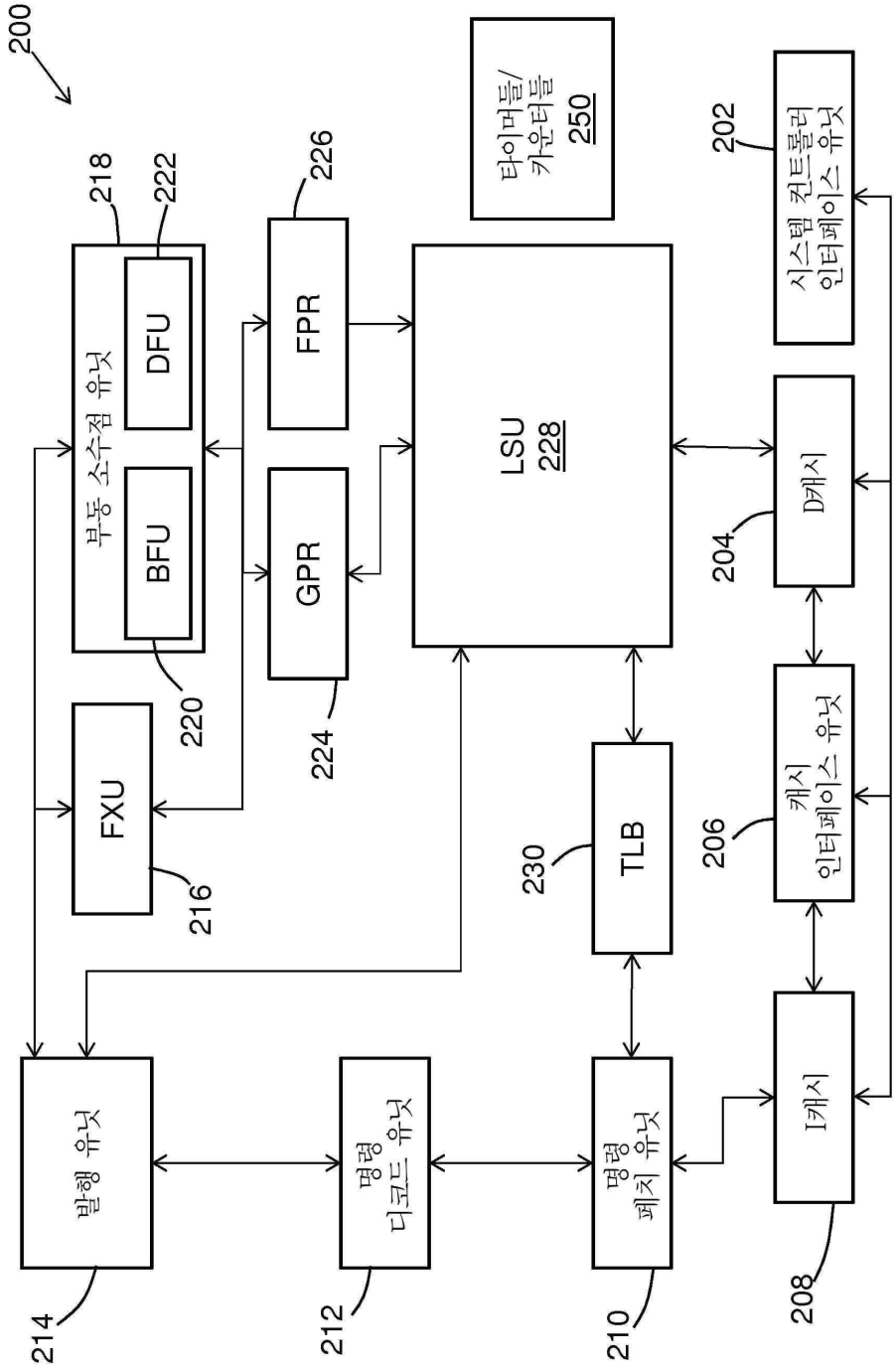
도면1a


100
입력/출력 서브시스템 104
시스템 메모리 160
하드웨어 시스템 영역 (HSA) 161
밀리코드 162
MT 퍼실리티 지원 코드 164
LPAR 하이퍼바이저 163
MT 퍼실리티 지원 코드 165
논리 파티션 1 170 (구성)
운영 체제 171
프로그램 172
프로그램 173
프로그램 174
프로그램 175
논리 파티션 2 180 (구성)
운영 체제 181
프로그램 182
프로그램 183
프로그램 184
논리 파티션 3 190 (구성)
운영 체제 191
프로그램 192
프로그램 193
프로세서 코어들 102
MT 퍼실리티 103
코어 1 110
코어 2 120
코어 3 130
코어 4 140
코어 5 150

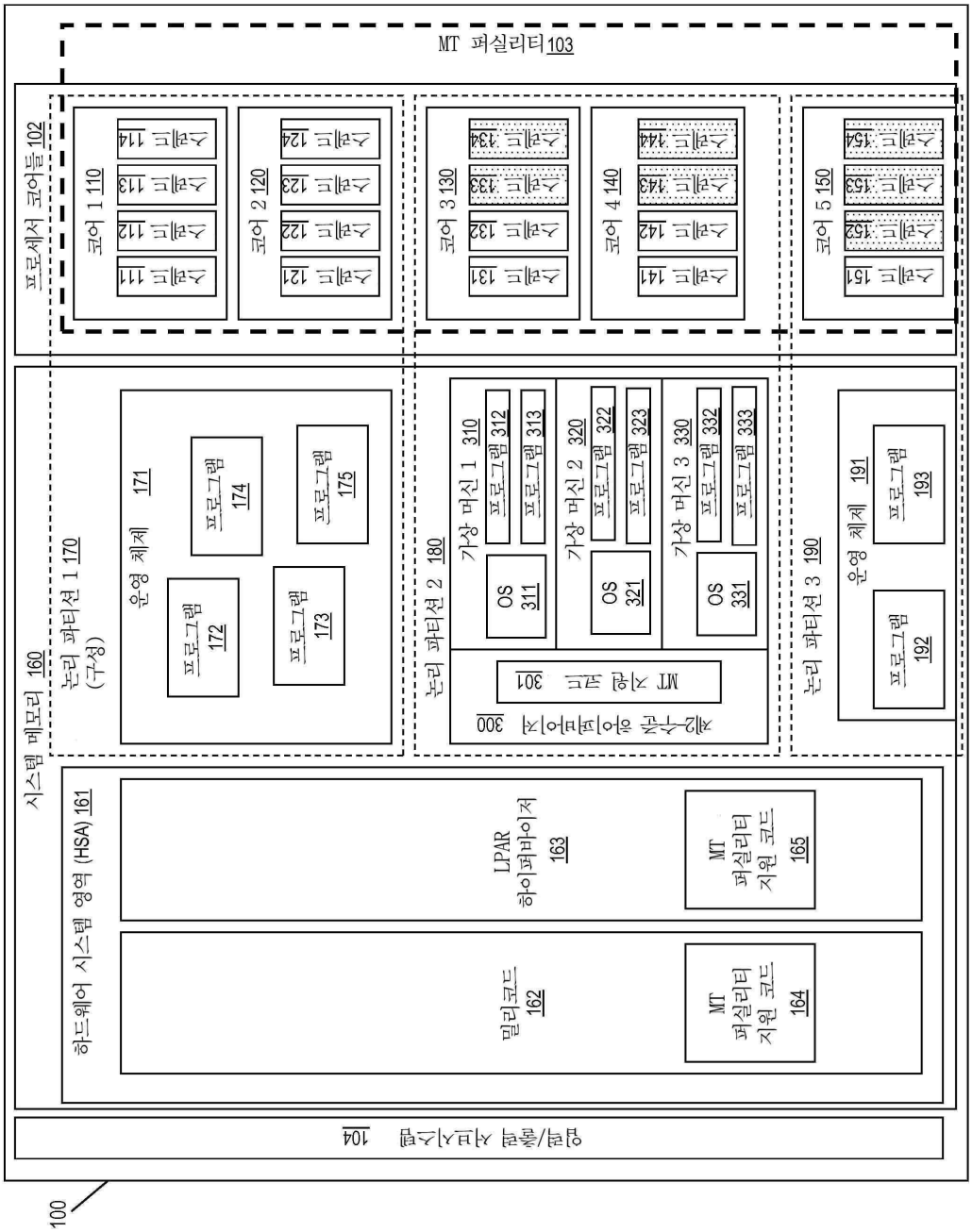
도면1b


100
입력/출력 서브시스템 104
시스템 메모리 160
프로세서 코어들 102
MT 퍼실리티 103
코어 1 110
코어 2 120
코어 3 130
코어 4 140
코어 5 150
하드웨어 시스템 영역 (HSA) 161
펌리코드 162
LPAR 하이퍼바이저 163
MT 퍼실리티 지원 코드 164
MT 퍼실리티 지원 코드 165
논리 파티션 1 (구성) 170
운영 체제 171
프로그램 172
프로그램 173
프로그램 174
프로그램 175
논리 파티션 2 (구성) 180
운영 체제 181
프로그램 182
프로그램 183
프로그램 184
논리 파티션 3 (구성) 190
운영 체제 191
프로그램 192
프로그램 193

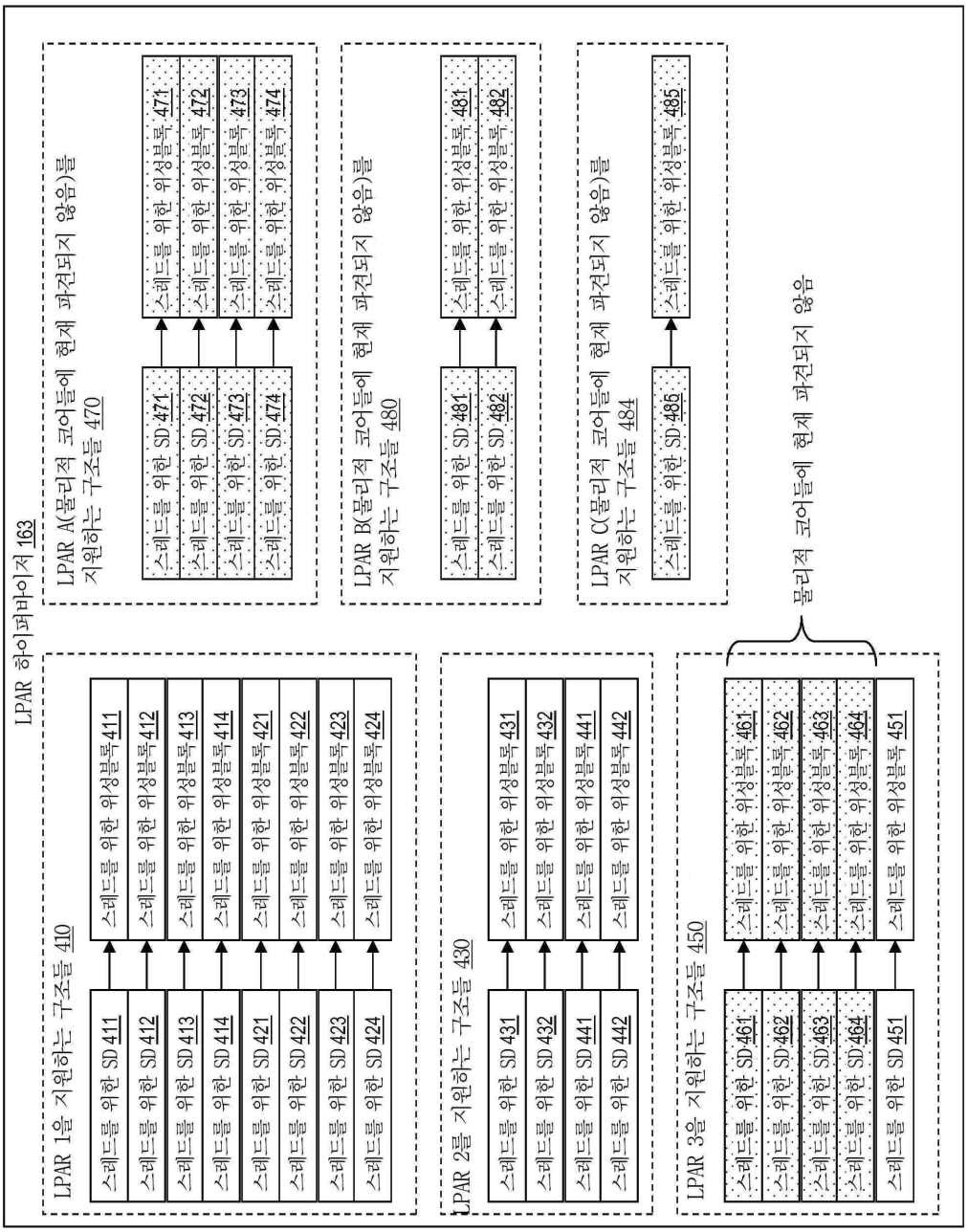
도면2


200
218
222
226
타이머들/
카운터들
250
202
시스템 컨트롤러
인터페이스 유닛
부동 소수점 유닛
DFU
BFU
FPR
LSU
228
D캐시
204
220
GPR
224
FXU
216
230
TLB
206
캐시
인터페이스 유닛
발행 유닛
명령
디코드 유닛
명령
페치 유닛
I캐시
214
212
210
208

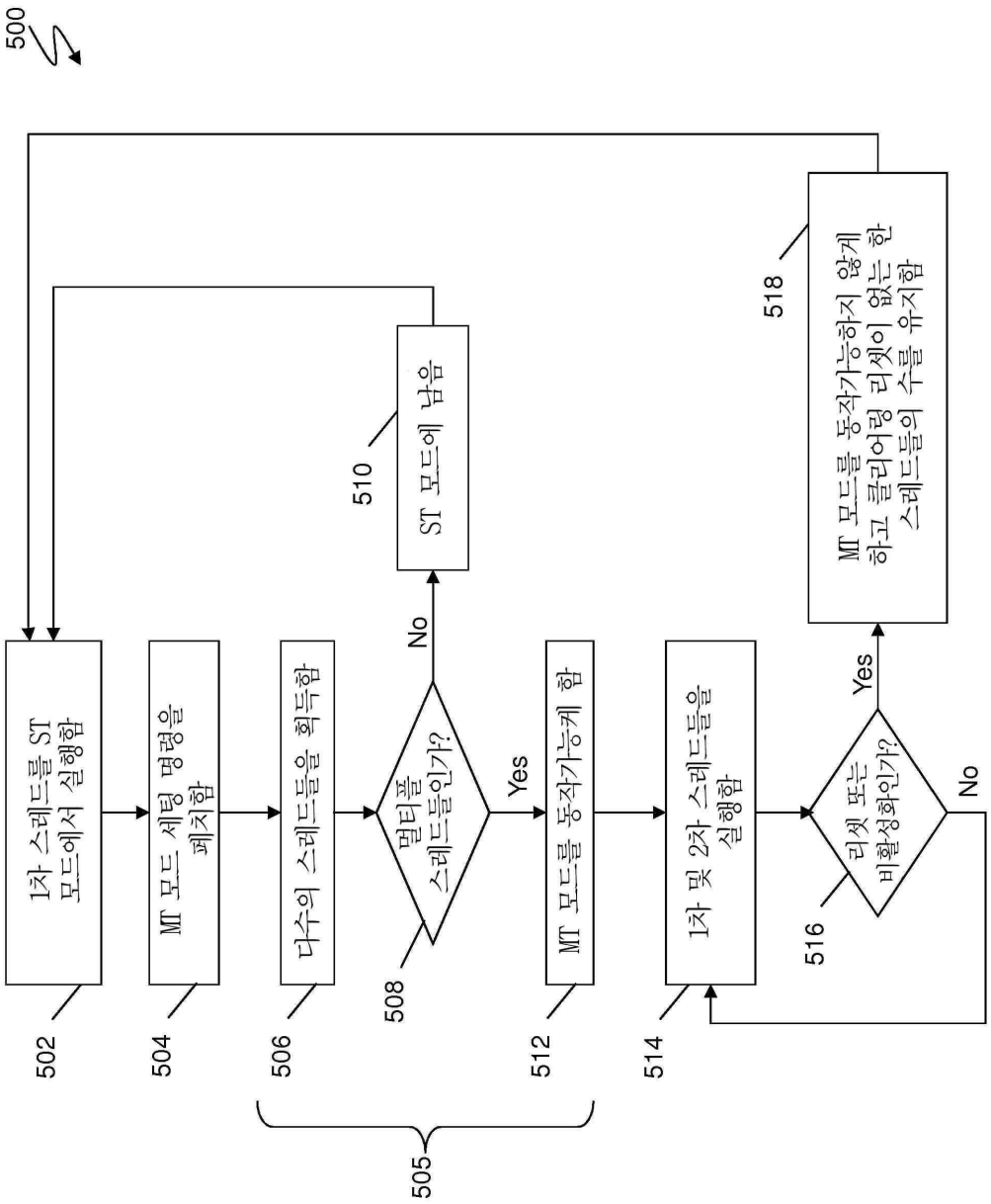
도면3


100
입력/출력 서브시스템 104
시스템 메모리 160
하드웨어 시스템 영역 (HSA) 161
펌리코드 162
LPAR 하이퍼바이저 163
MT 퍼실리티 지원 코드 164
MT 퍼실리티 지원 코드 165
논리 파티션 1 170 (구성)
운영 체제 171
프로그램 172
프로그램 173
프로그램 174
프로그램 175
논리 파티션 2 180
제1수준 하이퍼바이저 300
MT 지원 코드 301
가상 머신 1 310
OS 311
프로그램 312
프로그램 313
가상 머신 2 320
OS 321
프로그램 322
프로그램 323
가상 머신 3 330
OS 331
프로그램 332
프로그램 333
논리 파티션 3 190
운영 체제 191
프로그램 192
프로그램 193
프로세서 코어들 102
MT 퍼실리티 103
코어 1 110
코어 2 120
코어 3 130
코어 4 140
코어 5 150

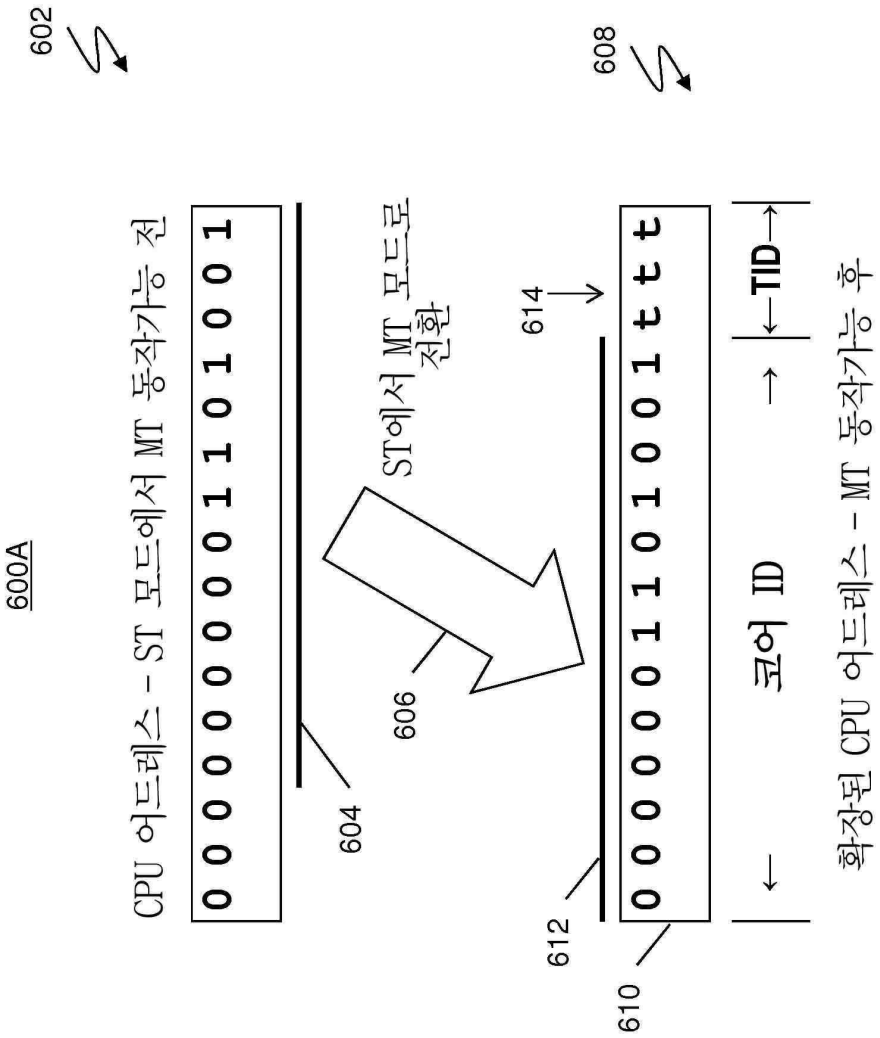
도면4


LPAR 하이퍼바이저 163
LPAR 1을 지원하는 구조들 410
스레드를 위한 SD 411
스레드를 위한 SD 412
스레드를 위한 SD 413
스레드를 위한 SD 414
스레드를 위한 SD 421
스레드를 위한 SD 422
스레드를 위한 SD 423
스레드를 위한 SD 424
스레드를 위한 위성블록 411
스레드를 위한 위성블록 412
스레드를 위한 위성블록 413
스레드를 위한 위성블록 414
스레드를 위한 위성블록 421
스레드를 위한 위성블록 422
스레드를 위한 위성블록 423
스레드를 위한 위성블록 424
LPAR 2를 지원하는 구조들 430
스레드를 위한 SD 431
스레드를 위한 SD 432
스레드를 위한 SD 441
스레드를 위한 SD 442
스레드를 위한 위성블록 431
스레드를 위한 위성블록 432
스레드를 위한 위성블록 441
스레드를 위한 위성블록 442
LPAR 3을 지원하는 구조들 450
스레드를 위한 SD 461
스레드를 위한 SD 462
스레드를 위한 SD 463
스레드를 위한 SD 464
스레드를 위한 SD 451
스레드를 위한 위성블록 461
스레드를 위한 위성블록 462
스레드를 위한 위성블록 463
스레드를 위한 위성블록 464
스레드를 위한 위성블록 451
물리적 코어들에 현재 파견되지 않음
LPAR A(물리적 코어들에 현재 파견되지 않음)를 지원하는 구조들 470
스레드를 위한 SD 471
스레드를 위한 SD 472
스레드를 위한 SD 473
스레드를 위한 SD 474
스레드를 위한 위성블록 471
스레드를 위한 위성블록 472
스레드를 위한 위성블록 473
스레드를 위한 위성블록 474
LPAR B(물리적 코어들에 현재 파견되지 않음)를 지원하는 구조들 480
스레드를 위한 SD 481
스레드를 위한 SD 482
스레드를 위한 위성블록 481
스레드를 위한 위성블록 482
LPAR C(물리적 코어들에 현재 파견되지 않음)를 지원하는 구조들 484
스레드를 위한 SD 485
스레드를 위한 위성블록 485

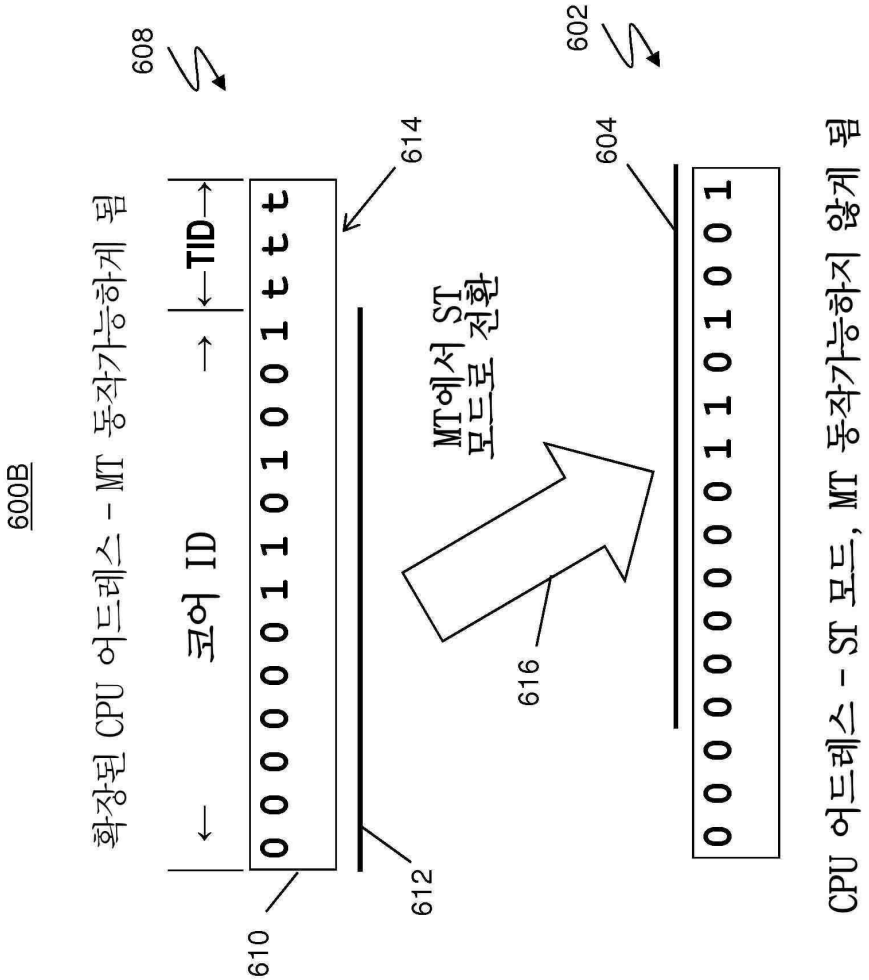
도면5


500
1차 스레드를 ST 모드에서 실행함
502
MT 모드 세팅 명령을 페치함
504
다수의 스레드들을 획득함
506
멀티플 스레드들인가?
508
No
ST 모드에 남음
510
Yes
MT 모드를 동작가능케 함
512
505
1차 및 2차 스레드들을 실행함
514
리셋 또는 비활성화인가?
516
Yes
No
MT 모드를 동작가능하지 않게 하고 클리어링 리셋이 없는 한 스레드들의 수를 유지함
518

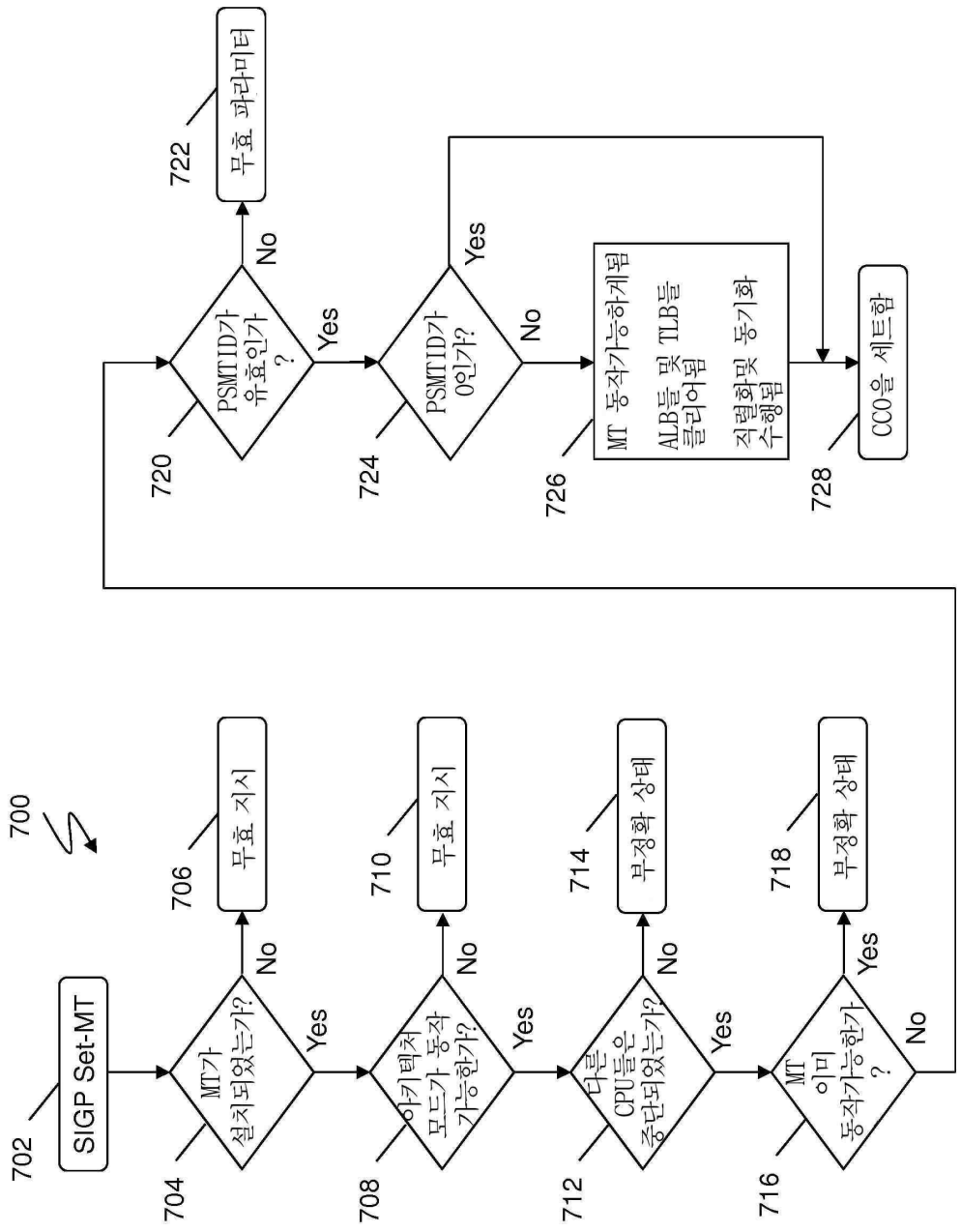
도면6a


600A
602
608
CPU 어드레스 - ST 모드에서 MT 동작가능 전
0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1
604
606
ST에서 MT 모드로
전환
614
612
610
0 0 0 0 0 0 0 0 1 1 0 1 0 0 1 t t t t
코어 ID
TID
확장된 CPU 어드레스 - MT 동작가능 후

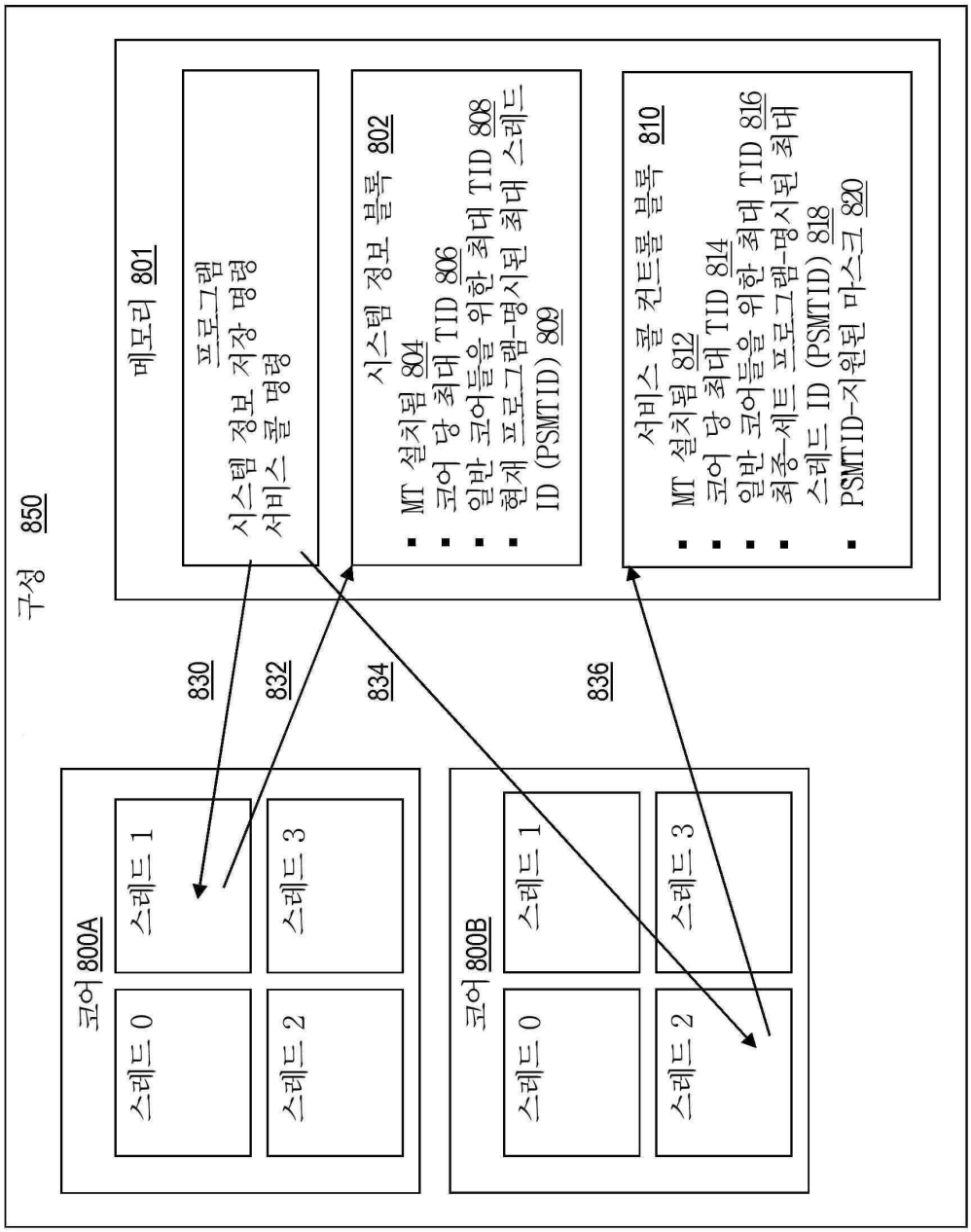
도면6b


600B
확장된 CPU 어드레스 - MT 동작가능하게 됨
608
코어 ID
TID
0 0 0 0 0 0 1 1 0 1 0 0 1 t t t t
614
610
612
MT에서 ST 모드로 전환
616
602
604
0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1
CPU 어드레스 - ST 모드, MT 동작가능하지 않게 됨

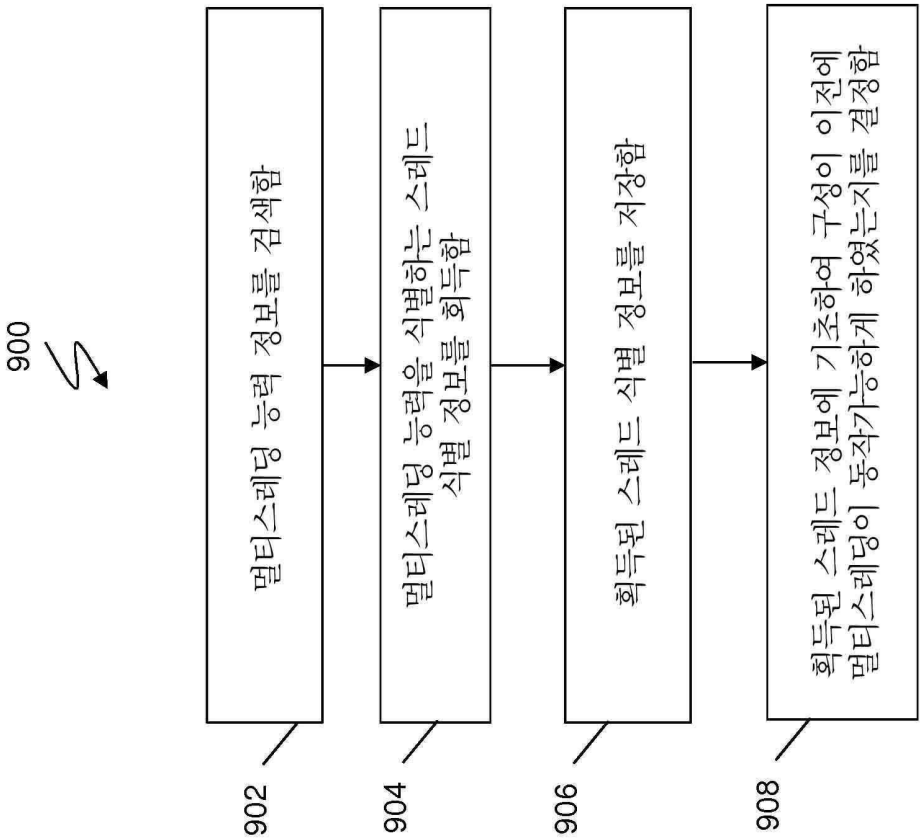
도면7


700
702
SIGP Set-MT
704
MT가 설치되었는가?
Yes
No
706
무효 지시
708
아키텍처 모드가 동작 가능한가?
Yes
No
710
무효 지시
712
다른 CPU들은 중단되었는가?
Yes
No
714
부정확 상태
716
MT 이미 동작가능한가 ?
Yes
No
718
부정확 상태
720
PSMTID가 유효인가 ?
Yes
No
722
무효 파라미터
724
PSMTID가 0인가?
Yes
No
726
MT 동작가능하게됨
ALB들 및 TLB들 클리어됨
직렬화및 동기화 수행됨
728
CCO을 세트함

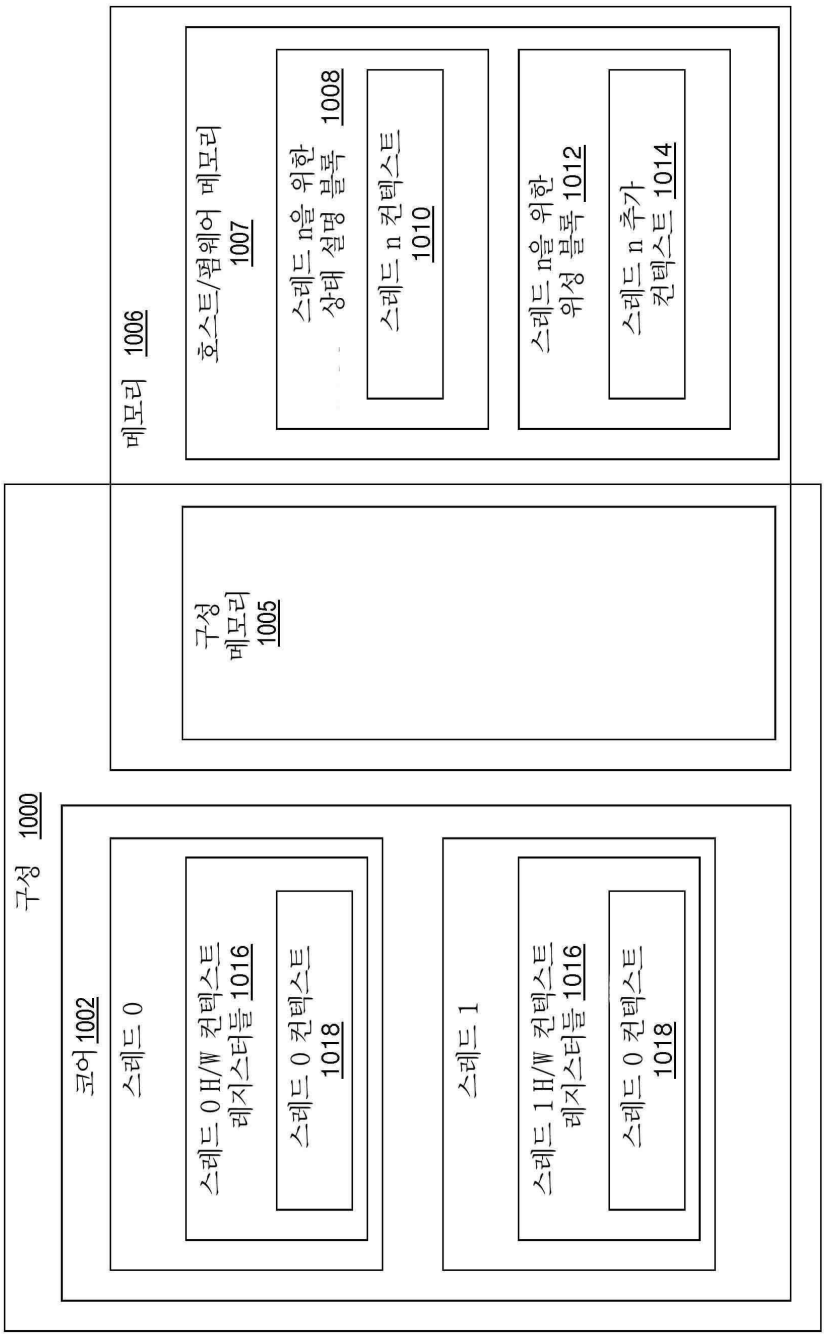
도면8


구성 850
메모리 801
프로그램
시스템 정보 저장 명령
서비스 콜 명령
시스템 정보 블록 802
MT 설치됨 804
코어 당 최대 TID 806
일반 코어들을 위한 최대 TID 808
현재 프로그램-명시된 최대 스레드 ID (PSMTID) 809
서비스 콜 컨트롤 블록 810
MT 설치됨 812
코어 당 최대 TID 814
일반 코어들을 위한 최대 TID 816
최종-세트 프로그램-명시된 최대 스레드 ID (PSMTID) 818
PSMTID-지원된 마스크 820
830
832
834
836
코어 800A
스레드 0
스레드 1
스레드 2
스레드 3
코어 800B
스레드 0
스레드 1
스레드 2
스레드 3

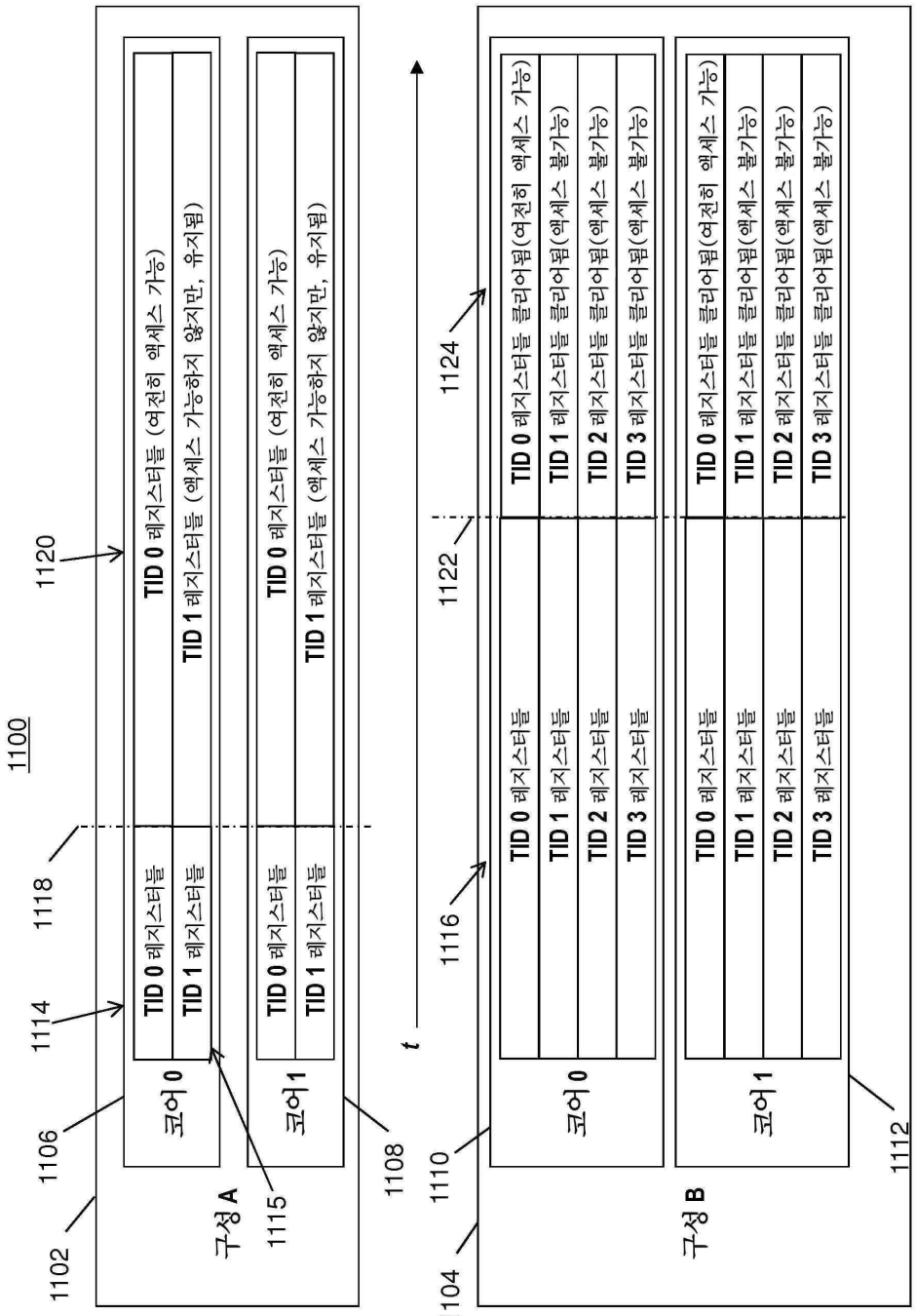
도면9


900
멀티스레딩 능력 정보를 검색함
902
멀티스레딩 능력을 식별하는 스레드 식별 정보를 획득함
904
획득된 스레드 식별 정보를 저장함
906
획득된 스레드 정보에 기초하여 구성이 이전에 멀티스레딩이 동작가능하게 하였는지를 결정함
908

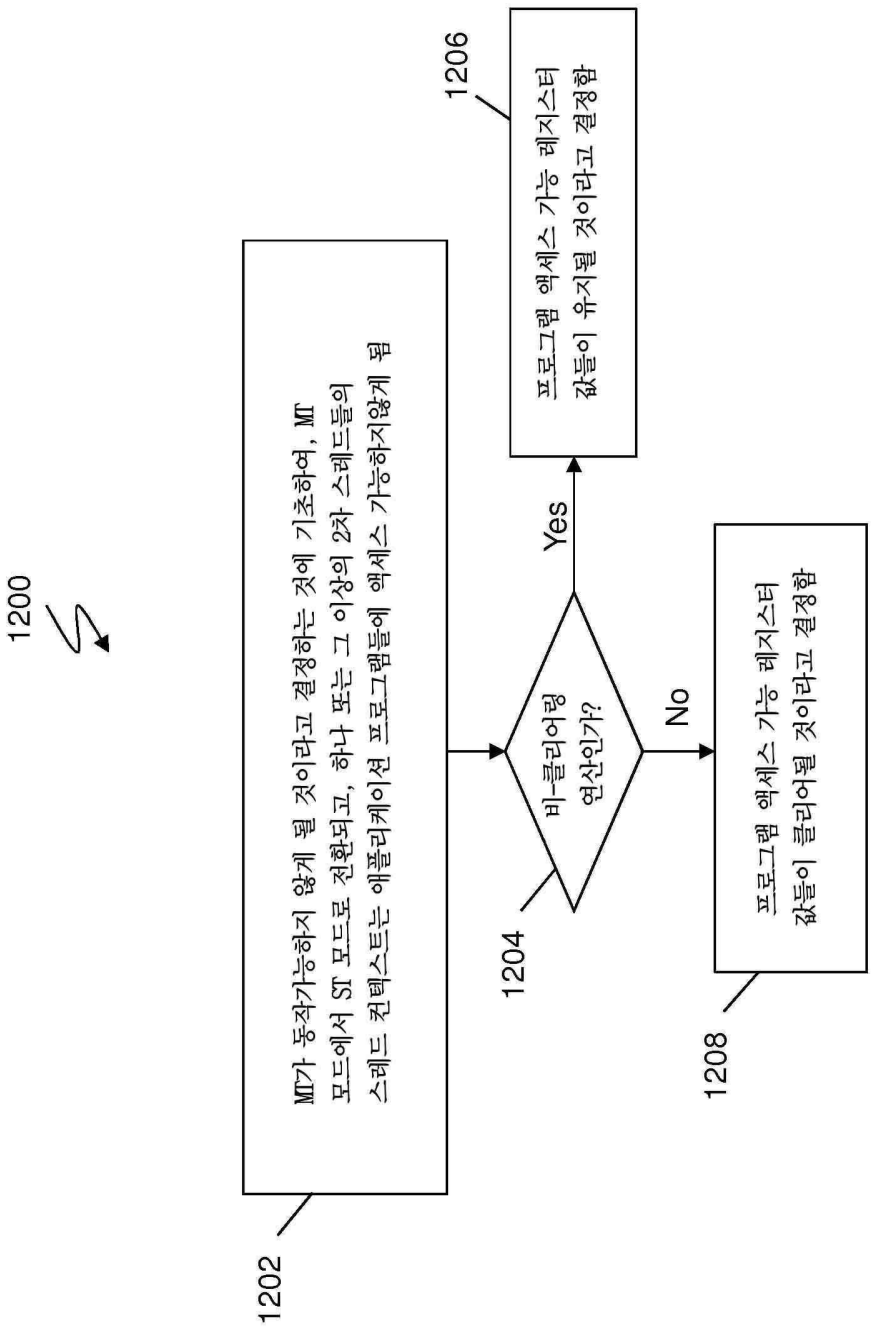
도면10


구성 1000
코어 1002
스레드 0
스레드 0 H/W 컨텍스트 레지스터들 1016
스레드 0 컨텍스트 1018
스레드 1
스레드 1 H/W 컨텍스트 레지스터들 1016
스레드 0 컨텍스트 1018
메모리 1006
구성 메모리 1005
호스트/펌웨어 메모리 1007
스레드 n을 위한 상태 설명 블록 1008
스레드 n 컨텍스트 1010
스레드 n을 위한 휘성 블록 1012
스레드 n 추가 컨텍스트 1014

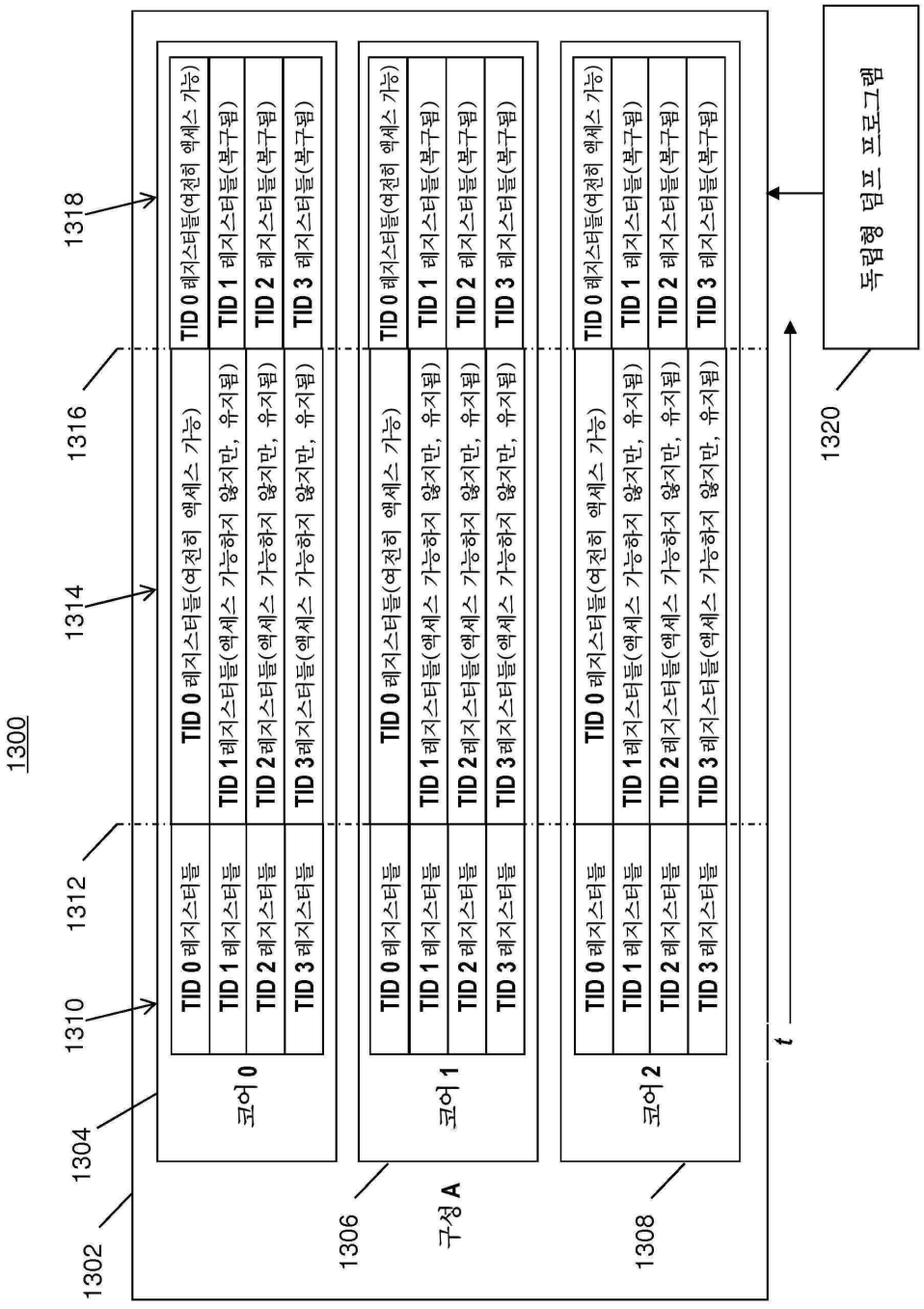
도면11


1100
1102
1104
1106
1108
1110
1112
1114
1115
1116
1118
1120
1122
1124
구성 A
구성 B
코어 0
코어 1
TID 0 레지스터들
TID 1 레지스터들
TID 0 레지스터들 (여전히 액세스 가능)
TID 1 레지스터들 (액세스 가능하지 않지만, 유지됨)
TID 0 레지스터들 클리어됨(여전히 액세스 가능)
TID 1 레지스터들 클리어됨(액세스 불가능)
TID 2 레지스터들
TID 3 레지스터들
TID 2 레지스터들 클리어됨(액세스 불가능)
TID 3 레지스터들 클리어됨(액세스 불가능)
t

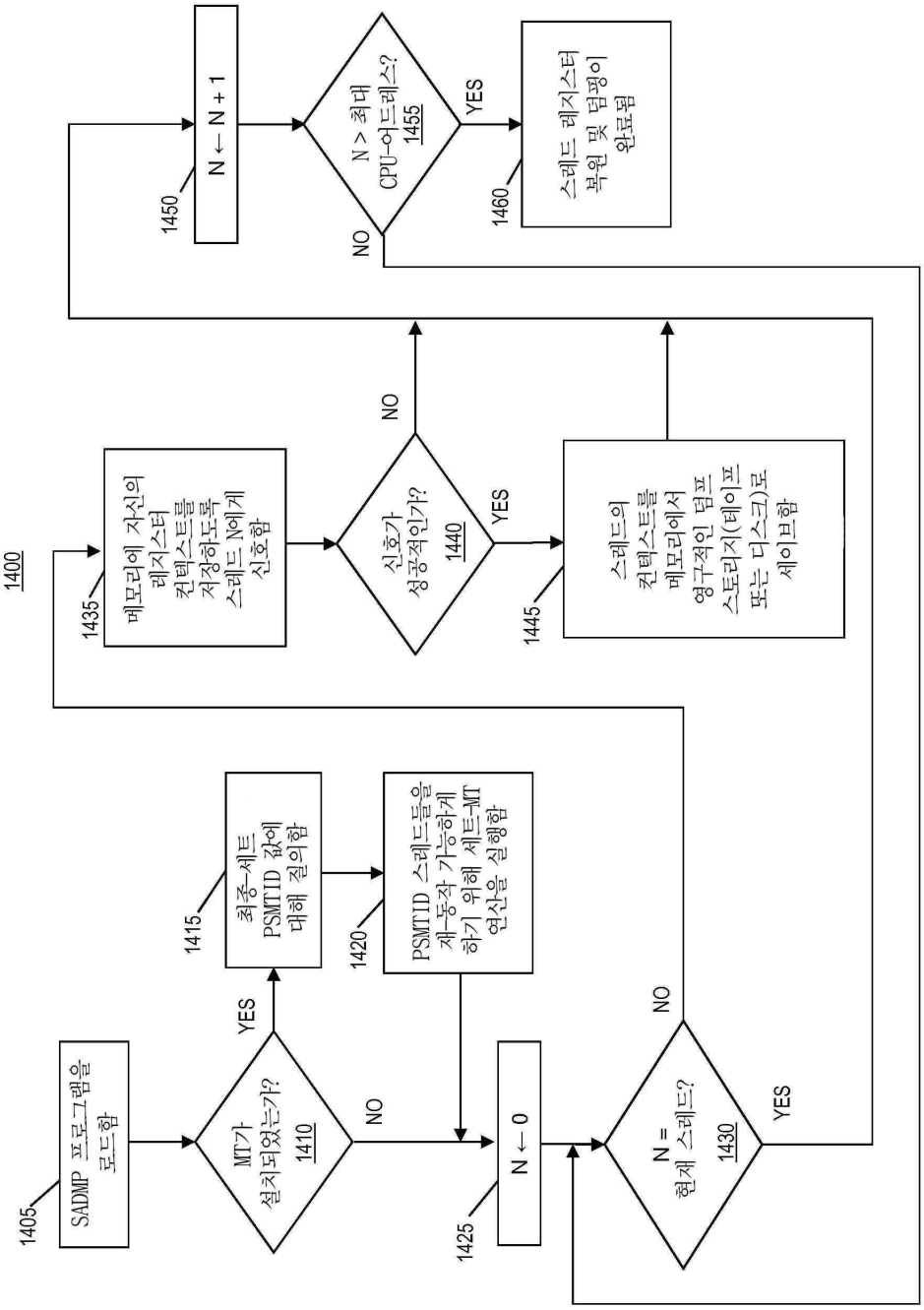
도면12


1200
MT가 동작가능하지 않게 될 것이라고 결정하는 것에 기초하여, MT
모드에서 ST 모드로 전환되고, 하나 또는 그 이상의 2차 스레드들의
스레드 컨텍스트는 애플리케이션 프로그램들에 액세스 가능하지않게 됨
1202
비-클리어링
연산인가?
1204
Yes
No
프로그램 액세스 가능 레지스터
값들이 유지될 것이라고 결정함
1206
프로그램 액세스 가능 레지스터
값들이 클리어될 것이라고 결정함
1208

도면13


1300
1302
1304
1306
1308
1310
1312
1314
1316
1318
1320
구성 A
코어 0
코어 1
코어 2
TID 0 레지스터들
TID 1 레지스터들
TID 2 레지스터들
TID 3 레지스터들
TID 0 레지스터들(여전히 액세스 가능)
TID 1 레지스터들(액세스 가능하지 않지만, 유지됨)
TID 2 레지스터들(액세스 가능하지 않지만, 유지됨)
TID 3 레지스터들(액세스 가능하지 않지만, 유지됨)
TID 0 레지스터들(여전히 액세스 가능)
TID 1 레지스터들(복구됨)
TID 2 레지스터들(복구됨)
TID 3 레지스터들(복구됨)
t
독립형 덤프 프로그램

도면14


1400
1405
SADMP 프로그램을 로드함
MT가 설치되었는가? 1410
YES
NO
1415
최종-세트 PSMTID 값에 대해 질의함
1420
PSMTID 스레드들을 재-동작 가능하게 하기 위해 세트-MT 연산을 실행함
1425
N ← 0
N = 현재 스레드? 1430
NO
YES
1435
메모리에 자신의 레지스터 컨텍스트를 저장하도록 스레드 N에게 신호함
신호가 성공적인가? 1440
NO
YES
1445
스레드의 컨텍스트를 메모리에서 영구적인 덤프 스토리지(테이프 또는 디스크)로 세이브함
1450
N ← N + 1
N > 최대 CPU-어드레스? 1455
NO
YES
1460
스레드 레지스터 복원 및 덤핑이 완료됨

도면15

