(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2014년05월07일
(11) 등록번호 10-1392109
(24) 등록일자 2014년04월28일

(51) 국제특허분류(Int. Cl.)
G06F 13/10 (2006.01)
(21) 출원번호 10-2012-7008044
(22) 출원일자(국제) 2010년08월02일
심사청구일자 2012년03월29일
(85) 번역문제출일자 2012년03월29일
(65) 공개번호 10-2012-0061938
(43) 공개일자 2012년06월13일
(86) 국제출원번호 PCT/US2010/044089
(87) 국제공개번호 WO 2011/025626
국제공개일자 2011년03월03일
(30) 우선권주장
12/550,737 2009년08월31일 미국(US)
(56) 선행기술조사문헌
KR1020080027006 A*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
인텔 코포레이션
미국 캘리포니아주 95054 산타클라라 미션 칼리지 불바드 2200
(72) 발명자
나투 마헤쉬 에스
미국 오레곤주 97229 포틀랜드 노스웨스트 바니스터 드라이브 5554
가네산 바스카란
인도 카 560037 방갈로르 에어포트 로드 136
(뒷면에 계속)
(74) 대리인
제일특허법인

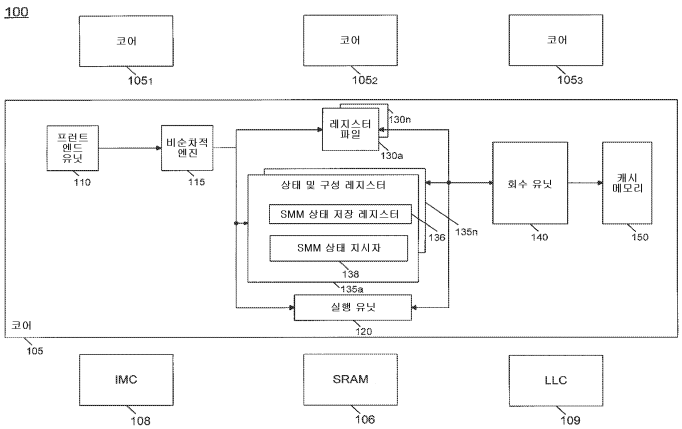
전체 청구항 수 : 총 11 항 심사관 : 고재용

(54) 발명의 명칭 **시스템 관리 모드의 프로세서에 상태 스토리지를 제공하기 위한 장치, 방법 및 시스템**

(57) 요 약

일실시예에 있어서, 본 발명은 시스템 관리 모드(SMM)로의 진입 시에 프로세서의 아키텍처 상태 스토리지로부터 교환되는 1개 이상의 아키텍처 상태를 저장하기 위해 정적 랜덤 액세스 메모리와 같은 온 다이 스토리지를 갖는 프로세서를 포함한다. 이와 같이 시스템 관리 메모리에 대한 이 상태 정보의 통신이 회피될 수 있어, SMM으로의 진입과 관련된 레이턴시를 감소시킨다. 또한, 실시예는 SMM 내부의 에이전트에 지시를 제공하기 위해 프로세서가 긴 명령 플로우에 있거나 또는 시스템 관리 인터럽트(SMI) 차단 상태에 있는 에이전트를 실행하는 상태를 업데이트하는 것을 가능하게 할 수 있다. 다른 실시예가 기재되고 청구된다.

대 표 도



(72) 발명자

란가라잔 타누나산

인도 방갈로르 560043 캄마나할리 메인 로드 잘 바유 비하르 에이-89

쿠마르 모한 제이

미국 오레곤주 97007 알로하 사우스웨스트 마코 레인 18680

도시 가우탐 비

인도 카 560008 방갈로르 에어포트 로드 다이아몬드 디스트릭트 오-73

파타사라시 라제쉬 에스

미국 오레곤주 97124 힐스보로 노스이스트 크릭세지 드라이브 1209

다타 샤만나 옘

미국 오레곤주 97124 힐스보로 노스이스트 레녹스 스트리트 532

빈스 프랭크

미국 오레곤주 97229 포틀랜드 노스웨스트 피나클 드라이브 2420

머시 라제쉬 나가라자

인도 카 560034 방갈로르 에이치에스알 레이아웃 섹터 - 이 22번 “비” 메인

스완슨 로버트 씨

미국 워싱턴주 98516 올림피아 그레이호크 레인 7142

특허청구의 범위

청구항 1

프로세서로서,

명령을 실행하고 시스템 관리 모드(SMM)에 진입하는 프로세서 코어를 포함하되, 상기 SMM으로의 진입 시에 상기 프로세서 코어는 상기 프로세서 코어의 구조적 상태 스토리지에 존재하는 활성 상태를 상기 구조적 상태 스토리지와 별개인 상기 프로세서의 저장 유닛 내에 저장하고 상기 구조적 상태 스토리지 내에 상기 SMM과 관련된 값을 삽입함으로써 SMM 핸들러의 실행을 위한 SMM 실행 환경을 설정하고, 상기 프로세서 코어는 SMM 코드에만 액세스가능한 기계 특정 레지스터(MSR)와 같은 상기 저장 유닛에 저장된 정보를 노출시키며,

상기 프로세서 코어가 상기 활성 상태를 시스템 관리 랜덤 액세스 메모리(SMRAM)가 아닌 상기 저장 유닛에 저장 가능함을 나타내는 지시자(indicator)를 저장하는 제 1 상태 레지스터를 더 포함하는

프로세서.

청구항 2

삭제

청구항 3

삭제

청구항 4

제 1 항에 있어서,

상기 제 1 상태 레지스터는 상기 SMM에서 실행되는 에이전트에 의해서만 업데이트될 수 있는

프로세서.

청구항 5

제 1 항에 있어서,

상기 프로세서 코어는 상기 SMRAM에 저장된 SMM 코드를 실행하는

프로세서.

청구항 6

제 5 항에 있어서,

상기 SMM이 메모리 에러를 복구할 때, 상기 프로세서 코어는 복구 SMM 코드를 비휘발성 메모리로부터 획득하고 상기 SMM 코드를 상기 SMRAM으로부터 획득하지 않는

프로세서.

청구항 7

제 1 항에 있어서,

상기 프로세서 코어의 논리 프로세서가 긴 플로우 동작(long flow execution) 중인 것을 나타내는 지시자를 저

장하는 제 2 상태 레지스터를 더 포함하는

프로세서.

청구항 8

제 7 항에 있어서,

상기 프로세서 코어의 논리 프로세서가 시스템 관리 인터럽트(SMI) 금지 상태에 있는 것을 나타내는 지시자를 저장하는 제 3 상태 레지스터를 더 포함하는

프로세서.

청구항 9

제 8 항에 있어서,

상기 SMM에 진입한 상기 프로세서 코어의 각 논리 프로세서의 지시자를 저장하는 SMM 지시자 맵을 더 포함하는

프로세서.

청구항 10

제 8 항에 있어서,

상기 제 1, 제 2 및 제 3 상태 레지스터는 상기 SMM 외에는 기록불가능한

프로세서.

청구항 11

제 1 항에 있어서,

상기 프로세서 코어의 논리 프로세서 모두가 상기 SMM에서 랑데뷰(rendezvous)를 수행한 상태가 아니어도 SMM을 실행하는 모나크(monarch) 프로세서를 더 포함하는

프로세서.

청구항 12

제 11 항에 있어서,

상기 모나크 프로세서는 상기 프로세서 코어의 논리 프로세서가 긴 플로우 동작 중인 것을 나타내는 제 1 상태 레지스터, 상기 프로세서 코어의 논리 프로세서가 SMI 금지 상태에 있는 것을 나타내는 제 2 상태 레지스터 및 상기 SMM에 진입한 상기 프로세서 코어의 각 논리 프로세서를 나타내는 SMM 지시자 맵에 액세스하고, 그에 기초하여 상기 논리 프로세서 모두가 랑데뷰를 수행한 상태가 아니어도 요청된 SMM 동작을 수행할 지 여부를 판단하는

프로세서.

청구항 13

제 12 항에 있어서,

상기 모나크 프로세서는 상기 프로세서 코어의 각 논리 프로세서가 상기 SMM에 진입했거나, 긴 플로우 동작에 있거나, SMI 금지 상태에 있다면 상기 논리 프로세서 모두가 상기 랑데뷰를 수행한 상태가 아니어도 요청된 SMM 동작을 수행하는

프로세서.

청구항 14

삭제

청구항 15

삭제

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

삭제

청구항 21

삭제

청구항 22

삭제

청구항 23

삭제

청구항 24

삭제

명 세 서

배 경 기 술

[0001] 대부분의 컴퓨터 시스템 프로세서는 시스템 관리 모드(SMM)라는 특수 동작 모드를 지원한다. SMM은 운영 시스템(OS) 소프트웨어에 투과성인 별개의 동작 환경을 제공한다. 이 모드는 시스템 관리, 장치, 전력 및 열 관리와 같은 특수 작업을 수행하도록 주문자 상표 부착 생산(OEM)에 의해 종종 사용된다. 서버 관련 신뢰성, 가용성 및 편리성(RAS) 기능은 통상 SMM을 사용하여 구현된다. SMM은 통상 시스템 관리 인터럽트(SMI) 메시지를 프로세서에 송신함으로써 진입된다. SMI를 인식할 시에, 프로세서는 SMM에 특히 할당된 시스템 관리 랜덤 액세스 메모리(SMRAM)로 지칭되는 시스템 메모리의 일부에, 또한 프로세서 저장 상태라는 현재의 프로세서 컨텍스트를 저장하고, SMRAM에 포함된 SMI 핸들러 코드를 실행한다. SMI 핸들러가 그 동작을 완료했을 때, 그것은 특수(SMM에서만 유효) 재개 명령을 실행해서, 프로세서가 저장된 프로세서 컨텍스트를 SMRAM로부터 리로딩하고 인터럽트된 작업

의 실행을 재개하게 한다.

[0002] 멀티프로세서 시스템에서, 일반적으로 SMI 메시지는 모든 프로세서에 브로드캐스트된다. SMI 핸들러는 이벤트를 핸들링하기 위해 SMM 모나크(monarch)로 지칭되는 1개의 프로세서를 선택한다. 이 프로세서는 SMI 이벤트를 핸들링하기 전에 모든 다른 프로세서가 SMM 내부에 랑데뷰될 때까지 대기한다. 넌-모나크 프로세서는 모나크가 이벤트 핸들링을 완료할 때까지 SMM에서 유지된다. SMM 이벤트가 핸들링되었을 때, 모나크는 SMM을 탈출시키기 위해 다른 프로세서에 신호를 보낸다. 이 동기화된 진입 및 탈출 동작은 2개의 병렬 환경(OS 및 SMM) 사이에서 어떤 자원 충돌을 방지하기 위해 구현된다. 즉, 동시에 일부 프로세서가 OS 환경에서 활성이고 나머지가 SMM 환경에서 활성이면, 그들이 공유 자원을 수정함으로써 서로의 동작을 방해할 수 있어, 시스템이 충돌되게 하는 것이 가능하다. 게다가, 어떤 SMM 이벤트는 특정 논리 프로세서 또는 논리 프로세서 세트에 의해서만 핸들링될 수 있다. 브로드캐스트는 모든 논리 프로세서가 SMI에 진입할 것이므로 이 조건이 항상 충족되는 것을 보증한다.

[0003] 따라서, 멀티프로세서 시스템에서의 SMI 핸들링은 복잡하고 모든 시스템 자원을 소비할 수 있어, 프로세서가 SMM에 있는 동안 다른 유용한 임무의 핸들링을 방해해서, 그것은 운영 시스템에 이용가능하지 않다.

도면의 간단한 설명

[0004] 도 1은 본 발명의 일실시예에 따른 프로세서의 블록도이다.

도 2는 멀티프로세서 시스템의 블록도이다.

도 3은 본 발명의 일실시예에 따른 방법의 흐름도이다.

도 4는 본 발명의 다른 실시예에 따른 방법의 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0005] 각종 실시예에 있어서, 온 다이 스토리지는 SMM 진입/탈출에 대하여 개별 스레드의 저장 상태를 저장하도록 외부 물리적 메모리를 이용하는 대안으로 사용될 수 있다. 대조적으로, 현재의 시스템은 SMM을 진입 및 탈출시키기 위해 외부 물리적 메모리에 의존한다. 이 시스템 RAM에 관한 SMM 의존성은 필요불가결한 응용에서의 스케일링, 성능 및 신뢰성 관련 제한이 되고, 본 발명의 실시예를 사용하여 회피될 수 있다. 여기에 사용된 바와 같이, 용어 "스레드"는 프로세스(예를 들어, 레지스터 파일 및 관련 구성 및 상태 레지스터)와 관련된 아키텍처 상태의 프로세서에 스토리지를 포함하는 하드웨어 스레드를 지칭할 수 있는 것에 주목하라. 여기에 사용된 바와 같이, 용어 "하드웨어 스레드"는 용어 "논리 프로세서"와 동의어로 사용된다. 각 프로세서 코어는 전용 아키텍처 상태 스토리지를 각각 갖지만 프런트 엔트 유닛, 실행 유닛 등과 같은 다른 코어 자원을 공유하는 다수의 논리 프로세서를 포함할 수 있다.

[0006] 다른 구현에 있어서, SMM 동안 어떤 활성 스레드의 저장 상태를 저장하도록 제공된 온 다이 스토리지는 저장 상태 스토리지를 위한 작은 전용 메모리의 역할을 하도록 프로세서 그 자체에서 온 다이 정적 RAM(SRAM) 또는 레지스터 파일일 수 있다. 일부 프로세서는 전력 관리와 특정 작업, 예를 들어 고급 구성 및 전력 인터페이스(ACPI) 상태(예를 들어, C6 상태 또는 다른 전력 관리 동작)에 따른 것과 같은 OS 관리 저전력 상태를 위한 온 다이 SRAM을 포함할 수 있다. 그러한 프로세서에서, 스레드당 단위로 분할되는 이 SRAM의 일부는 각 스레드의 SRAM 저장 상태를 위해 보존될 수 있다. 일례로서, 각 논리 프로세서는 SMM 저장 상태에 대한 SRAM 스토리지의 1 킬로바이트(KB)를 사용할 수 있다. 소정 프로세서가 SMM 저장 상태에 대한 이 SRAM의 양을 제공할 수 없으면, 실시예는 그것이 C6 플로우를 위해 보존된 SRAM을 이용할 수 있도록 구현될 수 있다. 이 경우에, SMM 내부의 C6/C7 전환은 SMM 상태 저장을 위한 공유된 SRAM 공간의 상호 배타적인 사용을 보증하기 위해 낮은 저전력 상태(예를 들어, C3)로 강등될 수 있다. 일부 프로세서는 C6 상태 저장을 위한 전용 SRAM을 구현하지 못하는 대신에 C6 상태 저장 동안 프로세서 상태를 저장하기 위한 최종 레벨 캐시(LLC)의 일부를 사용한다. 이 프로세서에서 SMM 저장 상태는 LLC에 저장될 수 있다.

[0007] 저장되면, 이 내부 SMM 저장 상태는 다른 방식으로 액세스될 수 있다. 예로서, 내부 상태는 스레드당 단위로 모델 고유 레지스터(MSR) 어드레싱, 또는 역 호환가능 메커니즘을 사용하여 액세스될 수 있다. 통상, 프로세서는 어떤 시스템 메모리 어드레스에서 SMM 저장 상태에 액세스할 수 있다. 역 호환가능 메커니즘은 논리 프로세서의 액세스를 이 레거시 메모리 어드레스에 트랩하고 그것을 적절한 SRAM 위치로 재지정하는 프로세서에 로직을 포함한다. 그러한 재지정은 기존 기본 입/출력 시스템(BIOS) 소프트웨어와의 절대적인 역 호환성이 필요해지면 구

현될 수 있다. 이 MSR은 SMM 모드에서만 판독 또는 기록될 수 있고 SMM 저장 상태와 관련된 제한을 따른다. 1개의 논리 프로세서가 다른 프로세서의 저장 상태에 액세스할 필요가 있으면, 이것은 소프트웨어 프로토콜을 통해 달성될 수 있다.

[0008] 어떤 실시예에 있어서, 전용 프로세서 식별자 리프(leaf)(예를 들어, CPUID 리프) 또는 그 필드 또는 기능 활성화 MSR(모델 고유 레지스터) 비트는 내부 SRAM의 사용을 가능하게 하기 위해 사용될 수 있다. 이제 도 1을 참조하면, 본 발명의 일시예에 따른 프로세서의 블록도가 도시되어 있다. 도 1에 도시된 바와 같이, 프로세서(100)는 다단 파이프라인 비순차적 프로세서일 수 있다. 프로세서(100)는 여기에 기재된 SMM 기술과 관련하여 사용되는 각종 특징을 예시하기 위해 비교적 간략화된 도면으로 도시되어 있다. 보여지는 바와 같이, 프로세서(100)는 복수의 프로세서 코어(105)를 포함하는 멀티 코어 프로세서일 수 있고 단일 반도체 다이 상에 형성될 수 있다. 도 1의 실시예에서 4개의 그러한 코어가 도시되어 있을지라도, 본 발명의 범위가 이와 관련하여 한정되지 않는 것을 이해하라. 도 1에서 더욱 보여지는 바와 같이, 추가적인 구성요소는 프로세서(100)에 제공될 수 있다. 예를 들어, 집적 메모리 제어기(IMC)(108)는 정적 랜덤 액세스 메모리(SRAM)(106)와 함께 제공될 수 있다. 앞서 논의된 바와 같이, 어떤 구현에서 이 메모리는 SMRAM에 다르게 저장될 본 발명의 실시예에 따른 컨텍스트 상태를 저장하도록 사용될 수 있다. 더욱이, 프로세서(100)는 모든 프로세서 코어 사이에서 공유되는 공유 캐시일 수 있는 최종 레벨 캐시(LLC)(109)를 포함할 수 있다.

[0009] 도 1에 도시된 바와 같이, 프로세서(100)는 실행될 매크로 명령을 페치하고 그것을 코어에서 나중에 사용하기 위해 준비하는데 사용될 수 있는 프런트 엔트 유닛(110)을 포함한다. 예를 들어, 프런트 엔트 유닛(110)은 마이크로 명령(μop) 스토리지뿐만 아니라 마이크로코드 스토리지와 함께 명령 프리페처, 명령 디코더, 및 트레이스 캐시를 포함할 수 있다. 명령 프리페처는 매크로 명령을 메모리로부터 페치하고 그것을 명령 디코더에 공급해서 그것을 프리미티브, 즉 프로세서에 의해 실행되는 μop로 디코드할 수 있다. 트레이스 캐시는 디코드된 μop를 취해서 그것을 프로그램 순서 시퀀스로 어셈블리할 수 있다. 물론, 추가적인 구성요소 및 특징은 프런트 엔트 유닛(110)으로 구현될 수 있다.

[0010] 프런트 엔트 유닛(110)과 실행 유닛(120) 사이에 마이크로 명령을 수신하고 그것을 실행을 위해 준비하는데 사용될 수 있는 비순차적(OOO) 엔진(115)이 결합되어 있다. 특히, OOO 엔진(115)은 논리 레지스터의 리네이밍(renaming)을 레지스터 파일(130a)과 같은 각종 레지스터 파일 내의 저장 위치 상으로 제공할 뿐만 아니라 마이크로 명령 플로우를 재순서화하고 실행에 필요한 각종 자원을 할당하기 위해 각종 버퍼를 포함할 수 있다. 레지스터 파일(130)은 정수 및 부동 소수점 연산을 위한 개별 레지스터 파일을 포함할 수 있다. 다수의 레지스터 파일(130a-n)이 다른 논리 프로세서 각각에 대해 제공될 수 있는 것을 주목하라. 추가적인 레지스터, 즉 상태 및 구성 레지스터(135)가 제공될 수도 있다. 보여지는 바와 같이, 각 레지스터(135a-n) 세트는 다른 논리 프로세서에 대한 것일 수 있다. 이 각종 레지스터는 스레드에 관한 정보 및 실행된 다른 명령을 제공할 뿐만 아니라 다른 동작 모드에 대한 코어를 구성하는데 사용될 수 있다.

[0011] 도 1에 도시된 예에서, 그러한 레지스터는 SMM 저장 상태 레지스터(136)를 포함할 수 있다. 각종 구현에서, 복수의 그러한 레지스터는 코어에 동작되는 소정 스레드와 각각 관련되어 제공될 수 있다. 앞서 논의된 바와 같이, 그러한 레지스터는 지시자, 예를 들어 스레드의 상태가 예를 들어 SMM으로 진입될 때 코어 자체 내에 저장되게 할 수 있는 인에이블 비트를 저장할 수 있다. 이 지시자가 인에이블되지 않으면, SMM로의 진입 시에, 스레드의 컨텍스트는 그 대신에 SMRAM에 저장될 것이다. 어떤 실시예에 있어서, 이 MSR은 다른 프로세서 특징을 제어할 수 있는 다른 비트를 포함할 수 있다. 어떤 실시예에 있어서, 지시자를 포함하는 이 레지스터 파일(135)은 SMM에서만 변경가능하게 되므로, 그것이 SMM 외부의 멀웨어 구성요소에 의해 악의적으로 변경되는 것을 보호해서, 시스템의 보안 및 강건성 둘 다를 증가시킬 수 있다.

[0012] 더욱 보여지는 바와 같이, 레지스터 파일(135)은 1개 이상의 SMM 상태 지시자 레지스터(138)를 포함할 수도 있다. 그러한 지시자 레지스터는 논리 프로세서가 SMM으로 진입되는 것이 금지될 때 또는 논리 프로세서가 긴 플로우 실행에 있는지를 지시하는 위치를 각각의 논리 프로세서가 가진 상태에서 비트맵 또는 비트 벡터의 형태일 수 있다. 일실시예에 있어서, 개별 레지스터는 각각의 그러한 지시를 위해 제공될 수 있다. 교대로, 단일 레지스터가 제공될 수 있으므로 논리적으로 결합된 지시자는 각각의 논리 프로세서에 대한 이 상태 중 하나의 존재를 지시하는데 사용될 수 있다. 이 레지스터의 사용에 관한 다른 상세한 설명이 이하 기재된다.

[0013] 도 1을 더욱 참조하면, 각종 자원은 다른 전용 하드웨어 중에서 예를 들어 정수, 부동 소수점, 및 단일 명령 다수 데이터(SIMD) 논리 유닛을 포함하는 실행 유닛(120)에 제공될 수 있다. 결과는 실행된 명령이 유효하게 회수되고 결과 데이터가 프로세서의 아키텍처 상태에 관련될 수 있는지, 또는 명령의 적절한 회수를 방해하는 1개

이상의 예외가 발생되는지를 판단하기 위해 동작될 수 있는 회수 유닛(140)에 제공될 수 있다.

[0014] 도 1에 도시된 바와 같이, 회수 유닛(140)은 본 발명의 범위가 이와 관련하여 한정되지 않을지라도 일실시예에서 저레벨 캐시(예를 들어, L1 캐시)일 수 있는 캐시 메모리(150)에 연결된다. 또한, 실행 유닛(120)은 캐시(150)(도 1에 도시되지 않은)에 직접 연결될 수 있다. 캐시 메모리(150)에서, 데이터 통신은 고레벨 캐시, 시스템 메모리 등으로 발생할 수 있다. 도 1의 실시예에서 고레벨이 도시되어 있을지라도, 본 발명의 범위가 이와 관련하여 한정되지 않는 것을 이해하라. 예를 들어, 다른 실시예는 순차적 프로세서로 구현될 수 있다.

[0015] SMM 저장 상태를 프로세서에 내부적으로 저장시킴으로써, 시스템의 신뢰성 및 강건성이 개선될 수 있다. 즉, 통상 SMRAM이 존재하는 외부 동적 랜덤 액세스 메모리(DRAM) 장치 세트인 물리적 메모리는 메모리 에러에 취약하다. 본 발명의 실시예가 없으면, SMM 동작은 이 외부 메모리로부터 실행되므로 에러 조건에 의존될 수 없다. 본 발명의 실시예를 사용하는 대신에, SMRAM 메모리 신뢰성은 에러를 핸들링할 때 SMI 핸들러를 비휘발성 공간으로부터 실행시킴으로써 개선될 수 있다. 예를 들어, SMM 핸들러는 메모리 에러를 핸들링하고 있는 BIOS 플래시 또는 외부 SRAM과 같은 보다 강건한 스토리지로부터 실행될 수 있다. 또한, SMM 저장 상태가 프로세서의 내부에 있을 때 이 스토리지의 아키텍처 상태는 MSR을 통해서만 소프트웨어 외부에 노출될 수 있다. SMM 코드가 "재개(RSM)" 명령을 실행한 후 기계 실행 상태를 복구시키는데 필요한 프로세서의 마이크로 아키텍처 상태는 이 내부 기계 상태를 적법하게 사용하지 않으므로 외부 소프트웨어에 노출될 필요가 없다. 또한, 이것은 악성 소프트웨어 코드가 민감한 마이크로 아키텍처 상태(그것처럼, 저장된 데이터 스토리지가 SMRAM에 있으면)에 액세스되지 않아서, 기계가 보다 안전하고 강건하게 되는 것을 의미한다.

[0016] 또한, 실시예는 성능 및 레이턴시(latency)를 개선할 수 있다. 다수의 서버 응용/운영 시스템은 비균일 메모리 아키텍처(NUMA) 최적화되고 BIOS는 통상 메모리를 구성해서 인접한 메모리 범위인 전체 SMRAM은 단일 소켓에 맵핑된다. 그러므로, 모든 SMM 저장 상태/복구 상태 동작은 SMRAM이 로컬인 하나의 소켓에 포함된 것을 제외하고 모든 논리 CPU에 대한 원격 기록/원격 판독으로 나타날 것이다. 4개의 소켓과, 12개의 코어를 각각 갖는 서버 구성에 대한 성능 분석은 SMM 저장 상태 기록 동작이 인터커넥트 및 메모리 대역폭에 의해 한정될 수 있고 5 마이크로초까지 취할 수 있는 것을 지시한다. 응용이 보다 NUMA 최적화되므로, 프로세서는 원격 트래픽에 대한 보다 적은 버퍼를 할당할 수 있다. 그것이 발생할 때, SMRAM 저장 상태 기록 및 판독 동작은 한층 더 긴 시간을 취할 것이다. 운영 시스템은 통상 허용가능한 실시간 성능을 유지하고 고속 네트워크 링크에 관한 타임아웃을 회피하기 위해 CPU가 SMM에 얼마나 있는지를 제한한다. 이 제한의 초과는 OS 응답성, 응용 레이턴시에 영향을 미치고 운영 시스템 고장을 심지어 초래할 수 있다. 따라서, 본 발명의 실시예에 따른 온 다이 SMM 저장 상태를 사용하는 것은 레이턴시를 감소시키므로 추가 시간이 SMM 이벤트(SMM의 유용한 작업)를 서비스하는 SMM 핸들러에 대해 할당될 수 있게 한다.

[0017] 추가적으로, 실시예는 확장성을 개선할 수 있다. 멀티프로세서 시스템에서, SMI가 발생될 때, 시스템 내의 모든 스레드는 시스템 부트 동안 시스템 BIOS에 의해 정의되고 보존되는 바와 같이 외부 시스템 메모리 내의 그 자체 전용 저장 상태 영역에 그 저장 상태를 저장시켜야 한다. 시스템 내의 모든 스레드의 모든 저장 상태를 캡처하는데 필요한 SMRAM 공간으로 보존될 물리적 메모리의 전체 양은 시스템 내의 스레드에 따라 선형으로 증가한다. 대칭 멀티 스레딩 지원을 갖는 멀티 코어, 멀티 소켓 시스템에 대해서는, 공간량이 상당히 커질 수 있다(일실시예에서 대략 256 KB와 비슷할 수 있다). SMM 저장 상태를 위해 온 다이 스토리지를 제공함으로써, 모든 코어 및 그 스레드를 수용하기 위해 항시 확장되는 SMRAM 영역에 대한 요구가 회피될 수 있음으로써 스케일을 용이하게 한다. 또한, 그것은 모든 스레드에 대한 SMRAM에서 유일한 비중첩 영역을 찾아서 할당하는 BIOS의 필요성을 제거한다. 더욱이, 이것은 또한 메모리 보호 영역이 실리콘으로 구현된 것을 보존한다. 핫 플러그 개요에서, SMRAM 내의 구조적으로 정의된 SMM 저장 상태 영역은 1MB보다 작다. 본 발명의 실시예가 없으면, BIOS는 새로운 프로세서를 추가할 때 OS 공격 및/또는 간섭을 회피하기 위해 메모리 보호 범위를 설정하고 데이터를 이동시킨다. 실시예는 저장된 상태가 OS 표시 메모리에 더 이상 저장되지 않기 때문에 이것을 행할 필요를 제거한다.

[0018] 이제 도 2를 참조하면, 본 발명의 일실시예에 따른 멀티프로세서 시스템의 블록도가 도시되어 있다.

[0019] 도 2에 도시된 바와 같이, 멀티프로세서 시스템(200)은 복수의 프로세서(200_1 - 210_n) 총칭적으로 프로세서(210)를 포함한다. 도 2의 실시예에서 4개의 그러한 프로세서가 도시되어 있을지라도, 본 발명의 범위가 이와 관련하여 한정되지 않는 것을 이해하라. 도 2에 도시된 구현에서, 비균일 메모리 아키텍처(NUMA) 시스템은 시스템 메모리(220_1 및 220_3)가 인터커넥트(217_1 및 217_3)를 통해 프로세서(210_1 및 210_3)에 국부적으로 부속되도록 제공된다. 따라서, 프로세서(210_2 및 210_n)에 의한 메모리로의 액세스는 복수의 점 대 점(PTP) 인터커넥트(215) 중

하나와 프로세서(210_1 및 210_3) 중 하나를 통한 통신을 필요로 한다. 도 2의 구현에서 보여지는 바와 같이, DRAM일 수 있는 메모리(220_1)는 SMRAM(225)을 포함한다. 이 NUMA 아키텍처에서, SMRAM(225)은 전체 시스템에 대한 시스템 관리 저장 장치이다. 따라서, 본 발명의 실시예가 없으면, SMM 진입 또는 탈출에 관한 각 프로세서는 컨텍스트를 이 SMRAM(225)에 저장/복구할 필요가 있다. 이것은 차례로 SMM으로 진입 및 SMM으로부터 탈출을 위한 레이턴시를 증가시킬뿐만 아니라 PTP 인터커넥트(215) 및 인터커넥트(217_1) 상의 다량의 대역폭 사용을 야기시킨다.

[0020] 따라서, 각종 실시예에 있어서 각 프로세서(210)는 1개 이상의 코어(212) 및 집적 메모리 제어기(214)에 더하여 SRAM(216)을 포함할 수 있다. 각종 실시예에 있어서, SRAM(216)은 SMM 저장 상태의 스토리지를 위해 전용될 수 있다. 즉, 시스템 관리 인터럽트가 발생될 때, 각 프로세서(210)의 각종 논리 프로세서에 대한 컨텍스트 상태는 그 SRAM(216)에 국부적으로 저장되어, SMRAM(225)과 상태 정보의 통신에 대한 요구를 회피할 수 있다. 다른 실시예에 있어서, 전용 온 다이 스토리지 대신에, 이 컨텍스트 상태는 예를 들어 레지스터 파일의 온 칩 레지스터 또는 캐시 메모리와 같은 다른 위치에 저장될 수 있다. 도 2의 실시예에서 이러한 특정 구현이 도시되어 있을지라도, 본 발명의 범위는 이와 관련하여 한정되지 않는다. 예를 들어, 실시예는 또한 균일 메모리 아키텍처 시스템에 사용될 수 있다.

[0021] 이제 도 3을 참조하면, 본 발명의 일실시예에 따른 방법의 흐름도가 도시되어 있다. 도 3에 도시된 바와 같이, 방법(300)은 상태 정보를 저장하기 위한 SMRAM에 액세스할 필요없이 SMM으로의 진입을 핸들링하기 위해 수행될 수 있다. 논의의 용이성을 위해 다수의 구현에서 다수의 스레드가 SMM으로 함께 진입할 수 있을지라도 단일 하드웨어 스레드만이 제공되는 것으로 가정된다는 것에 주목하라. 도 3에 보여지는 바와 같이, 방법(300)은 시스템 관리 인터럽트를 수신함으로써 개시될 수 있다(블록 310). 이 인터럽트의 수신 시에, 현재 활성 상태(예를 들어, 소정 하드웨어 스레드의)는 온 다이 스토리지에 저장될 수 있다(블록 320). 앞서 논의된 바와 같이, 이 온 다이 스토리지는 전용 SRAM, 다른 목적(예를 들어, 전력 관리 상태)에 사용되는 SRAM, 레지스터 스토리지, 온 다이 캐시 스토리지 등일 수 있다.

[0022] 도 3을 더 참조하면, 프로세서 상태는 예를 들어 프로세서 명세에 의해 정의된 바와 같이 SMM 진입 상태를 일치시키기 위해 수정된다(블록 330). 이 상태는 레지스터 파일의 초기값뿐만 아니라 각종 제어 및 구성 레지스터의 값을 포함한다. 따라서, 이 설정은 SMM 진입 상태와 관련된 소정값을 상태 스토리지로 로딩함으로써 SMM 핸들러에 적절한 SMM 실행 환경을 준비한다. SMM 상태가 설정되었을 때, 제어가 블록 340으로 이동되며, 여기서 SMM은 SMRAM으로부터의 코드 및 데이터를 사용하여 실행될 수 있다(블록 340). 따라서, 소망의 SMM 동작이 수행될 수 있다. 본 발명의 범위가 이와 관련하여 한정되지 않을지라도, SMM 동작의 예는 전력 관리 동작, 에러 핸들링 동작 등을 포함한다.

[0023] 그 다음, SMM 동작이 완료되었는지가 판단될 수 있다(다이아몬드 350). 그렇지 않으면, SMM에서의 실행이 계속될 수 있다. 완료되었다면, 프로세서는 재개 명령을 실행한다(블록 360). 이 명령의 결과로서, 이전 상태는 온 다이 스토리지로부터 프로세서의 레지스터로 다시 로딩될 수 있다(블록 370). 이어서, 프로세서는 활성 상태로 다시 복구되는 이 이전 상태에 대응하는 스레드의 실행을 재개할 수 있다(블록 380). 도 3의 실시예에서 이러한 특정 구현이 도시되어 있을지라도, 본 발명의 범위가 이와 관련하여 한정되지 않는 것을 이해하라. 예를 들어, 어떤 구현에서, SMM 동작을 SMRAM으로부터 실행하는 것보다는 오히려, 특히 SMM이 DRAM 에러와 같은 에러를 핸들링하고 있을 때, 실시예는 그 대신에 플래시 메모리와 같은 비휘발성 스토리지로부터 SMM 상태 정보, SMM 코드 및 데이터를 획득할 수 있다.

[0024] 상술한 바와 같이, 활성 상태의 실리콘 스토리지는 SMM 레이턴시를 감소시킬 수 있다. 실시예는 이제 논의되는 바와 같이 어떤 상황에서 SMM으로 더 신속한 진입을 가능하게 함으로써 레이턴시를 더욱 감소시킬 수 있다.

[0025] SMM 레이턴시는 프로세서가 단일 SMI 당 SMM 환경에 있는 지속 시간으로 정의된다. 전체 SMM 레이턴시, 프로세서 오버헤드 및 OEM BIOS 코드에 대한 2개의 주요 기여자가 존재한다. 이 레이턴시는 타임아웃 및 클록 드리프트와 같은 OS 환경 상의 사이드 효과를 회피하기 위해 제어 하에 유지되어야 한다. 장래 수요는 감소될 이 레이턴스를 필요로 해서, 그것은 실현하기 어려워진다. 현재, SMI 레이턴시는 대략 190 마이크로초 미만이 되도록 조정된다. 인터넷 포털 데이터 센터 및 유틸리티 컴퓨팅과 같은 새로운 사용 모델은 더 예측가능한 레이턴시를 응용으로부터 예상한다. 그 결과, OS 벤더는 SMM 레이턴시의 추가 감소를 요구하고 있다. 한편, 다른 기술은 시간에 따른 SMI 레이턴시를 증가시키는 잠재력을 갖는다. 예로서, 멀티 코어 프로세서에 대한 업계 푸시는 SMI 핸들러가 항시 증가하는 수의 프로세서 코어를 랑데뷰해야 하는 것을 의미한다. 또한, 새로운 SMM 기반 능력은

SMM 레이턴시에 부가적인 프레셔를 가한다. 예를 들어, 고성능 RAS 능력은 SMM에 의존한다. 게다가, 어떤 OEM은 자사 제품을 차별화하는 독자 전력 관리 능력을 수행하는 SMM을 이용한다. 다수의 OEM은 초당 8배까지 SMI를 생성하는 것으로 알려져 있다.

[0026] 어떤 명령 세트 아키텍처(ISA)는 라이트 백(write back)과 같은 명령을 포함하고 명령(예를 들어, wbinvd)을 무효화하며, 그것은 모든 캐시 라인을 무효화해서 다시 메모리에 라이트 백한다. 이 동작은 특히 큰 캐시 사이즈를 지원하는 프로세서에서 완료하는데 시간이 오래 걸릴 수 있으며, 예를 들어 10^3 내지 10^7 프로세서 사이클과 비슷할 수 있다. 게다가, SMI 응답이 지연될 수 있는 어떤 프로세서 상태(예를 들어, C3 및 C6 저 프로세서 상태)가 존재한다. 일괄적으로, 이 명령 및 프로세서 상태는 완료하기 위해 매우 긴 사이클 수(예를 들어, 10^3 클록과 비슷한)를 취할 수 있는 명령 또는 프로세스를 의미하는 것으로 정의되고 SMM으로의 진입을 지연시킬 수 있는 "긴 플로우" 상태로 지칭된다. 일실시예에 있어서, 5 마이크로초 이상까지 SMM 진입을 지연시키는 어떤 플로우는 긴 플로우로 지칭될 수 있다. SMM에 관해서는, 1개 이상의 논리 프로세서가 긴 플로우에 있으면, 그것은 SMM 진입을 지연시킨다.

[0027] 앞서 설명된 바와 같이, SMM 모나크는 모든 예상된 논리 프로세서가 SMM에 진입했을 때까지 대기한다. SMM으로의 진입 시에, 각 프로세서는 그것이 SMM에 진입한 것을 지시하는 SMRAM에 그 자체의 비트를 설정한다. 모나크는 모든 예상된 프로세서가 자신의 비트를 설정했을 때까지 대기한다. 1개 이상의 논리 프로세서가 긴 플로우에 있고 SMM에 늦게 진입할 때, 그것은 SMM 모나크를 지체시키므로 SMM 레이턴시를 증가시킨다. 게다가, 대기 스타트업 인터프로세서 인터럽트(WFS) 및 TXT 슬립 상태와 같은 어떤 아키텍처 상태가 존재하며, 여기서 SMI 이벤트는 금지된다. OS/BIOS가 1개 이상의 논리 프로세서를 SMI 금지 상태로 배치하면, 그것은 OS/BIOS가 이 상태로부터 명시적으로 그것을 가져올 때까지 SMM에 진입하지 못할 것이다. SMI 이벤트는 모든 다른 프로세서를 SMM에 배치하므로, OS는 SMI를 노출시킬 수 없다. 이 개요 하에, SMM 모나크는 SMI 금지 프로세서의 존재를 결정하기 위해 긴 타임아웃에 의존해야 한다. 이 타임아웃은 SMM 랑데뷰를 지연시키고 전체 SMM 레이턴시를 증가시키거나 SMM 이벤트 핸들링에 이용가능한 시간량을 감소시킨다.

[0028] 각종 실시예에 있어서, SMM 내에서 타임아웃에 대한 요구는 어떤 논리적 프로세서가 긴 플로우에 심지어 있을지라도 회피될 수 있다. 그러한 타임아웃을 제거하는 것은 평균 SMM 레이턴시를 10-20%까지 그리고 최악의 경우 SMM 레이턴시를 적어도 몇 밀리초까지 개선할 수 있다.

[0029] 실시예는 긴 플로우 또는 SMI 금지 상태에 있는 프로세서가 공유 자원에 액세스될 것 같지 않은 사실에 의존한다. 게다가, 그러한 프로세서는 SMI를 발생시킬 것 같지 않으므로, 그 참여는 SMI 처리에 필요없다. 그러므로, SMM 모나크는 그러한 프로세서가 SMM으로 진입되기 전에 SMM을 속행할 수 있다.

[0030] 그러나, 속행 전에 SMM 모나크는 프로세서가 긴 플로우 및/또는 SMI 금지 상태에 있는 것을 확실히 검출할 수 있어야 한다. 긴 플로우 또는 SMI 금지 상태에서 비지(busy)한 프로세서를 검출하기 위해, 실시예는 지시자를 이 상태에 예를 들어 비트맵으로서 제공할 수 있다. 일실시예에 있어서, 그러한 지시는 LONG_FLOW_INDICATION 및 SMI_INHIBITED_INDICATION이라는 글로벌하게 보여지는 구성 레지스터를 통해 제공될 수 있다. 이 실시예에 있어서, 1개의 비트는 소켓에서 각 논리 프로세서에 할당될 수 있다. 일례로서, 레지스터는 도 1의 레지스터(138)에 의해 표시될 수 있다. 프로세서 마이크로코드가 긴 플로우 및 SMI 금지 상태로의 진입 및/또는 긴 플로우 및 SMI 금지 상태로부터 탈출에 수반되는 구현에서, 마이크로코드/하드웨어는 이 레지스터 비트를 파퓰레이트(populate)할 수 있다. 긴 플로우 중 일부는 5 마이크로초보다 긴 시간을 초래할 수 있으므로 이 상태에서 프로세서를 대기시키지 않는 능력은 SMM 레이턴시에 상당한 절약을 제공할 수 있다. 장래의 프로세서는 SMM 마이크로코드 진입 플로우에 대해 5 이상의 마이크로초가 걸릴 수 있고 긴 플로우 그 자체로 간주될 수 있다. SMM 모나크는 모든 프로세서가 설명되며, 즉 그것이 SMM에 조인하거나 긴 플로우 또는 SMI 금지 상태에 있는 것으로 보고될 때까지 대기할 수 있다. 그러한 결정을 원조하기 위해, SMRAM에 저장된 비트맵과 같은 1개 이상의 테이블이 이하에 기재되는 바와 같이 사용될 수 있다.

[0031] 하나의 구현에서, 모나크 프로세서는 지시자 레지스터의 체크를 수행하기 전에 그 상태를 저장하고 SMM 프리앰블 코드를 실행시킨다. 이 단계는 쉽게 0.5 이상의 마이크로초가 걸릴 수 있다. 이 지속 시간은 어떤 인-플라이트(in-flight) 인터럽트에 대한 전달 시간보다 훨씬 더 커서, 코어로의 SMI 전달과 그 지시자 레지스터의 판독 사이에 어떤 레이스 조건이 없다는 것을 보증한다. 지연이 어떤 구성 하에 보다 작으면, 모나크 프로세서는 작은 지연 루프를 삽입해서 구성할 수 있다.

[0032] 이제 도 4를 참조하면, 본 발명의 다른 실시예에 따른 방법의 흐름도가 도시되어 있다. 구체적으로, 도 4는 모

든 논리 프로세서가 SMM 상태에 랑데뷰될 필요가 없을 때 SMM으로의 진입 및 SMM으로부터의 탈출을 핸들링하기 위한 흐름도를 도시한다. 이와 같이, SMM 동작을 수행하기 전에 모든 논리 프로세서를 대기시키는 것과 관련된 레이턴시가 회피될 수 있다. 도 4에 보여지는 바와 같이, 방법(400)은 SMI 이벤트의 생성에 의해 개시될 수 있다(블록 410). 이 SMI 이벤트는 모든 스레드에 전달될 수 있다. 논의의 용이함을 위해 구현이 다수의 소켓을 통해 SMM을 랑데뷰하는데 사용될 수 있을지라도 도 4의 스레드가 단일 프로세서 소켓에 관한 것으로 가정된다는 것에 주목하라.

[0033] 그 다음, 지시자는 SMM 랑데뷰 상태에 진입하는 각 스레드에 대한 SMM 지시자 맵에 설정될 수 있다(블록 420). SMM에 진입하는 각종 예비 행위는 먼저 도 3과 관련하여 상술한 것과 같은 스레드, 예를 들어 상태 저장에 의해 수행될 수 있는 것이 이해되어야 한다. SMM 랑데뷰 상태에 진입하는 각 스레드는 SMRAM에 저장될 수 있는 SMM 지시자 맵에 지시자를 설정할 수 있다. 일실시예에 있어서, 이 맵은 각 논리 프로세서가 맵의 비트와 관련되는 비트 맵일 수 있고, 여기서 각 소켓의 논리 프로세서는 맵의 다른 세그먼트로 분리될 수 있다. 따라서, 소정 스레드가 SMM으로 진입될 때, 비트 맵 내의 그 대응하는 비트가 설정될 수 있다. 그 다음, SMM 내의 스레드 중 하나는 모나크 또는 실행 스레드로 선택될 수 있다(블록 430). 각종 실시예에 있어서, 그 스레드가 실행 스레드이어야 하는 결정이 변경될 수 있다. 예를 들어, 모나크는 사전 선택될 수 있거나(예를 들어, 소켓 0 상의 논리 프로세서 0) 선정 메커니즘을 통해 동적으로 선택될 수 있다.

[0034] 도 4를 더 참조하면, 각 스레드는 이 때 그것이 모나크로 선택되었는지를 판단할 수 있다(다이아몬드 435). 그렇지 않다면, 스레드는 슬립 상태로 진입될 수 있으며, 여기서 그것은 신호 완료에 대한 모나크 스레드를 대기시킨다(블록 470).

[0035] 따라서, 제어는 모나크 스레드를 위해 블록 440으로 이동된다. 거기서, 그것은 모든 스레드에 대한 ACCOUNTED 상태를 결정할 수 있다. 일실시예에 있어서, 이 상태는 각종 구성 레지스터에 기초할 수 있고, SMM 지시자 맵은, 스레드 존재 맵에 더하여, SMRAM에도 존재할 수 있다. 이 존재 맵은 SMM 지시자 맵과 유사한 맵일 수 있고 시스템에 존재하는 스레드를 지시하도록 SMM 초기화 동안 설정될 수 있다. 일실시예에 있어서, 블록 440에서의 판단은 다음과 같이 비트와이즈 OR 연산일 수 있다: OR(LONG_FLOW_INDICATION,SMI_INHIBITED_INDICATION, IN_SMM_INDICATION)

[0036] 여기서 LONG_FLOW_INDICATION은 비트 벡터를 저장하는 상태 레지스터로부터 획득되며, 그 각각의 비트는 대응하는 스레드가 긴 플로우 동작에 있는지를 지시하며, SMI_INHIBITED_INDICATION은 비트 벡터를 저장하는 상태 레지스터로부터 획득되며, 그 각각의 비트는 대응하는 스레드가 SMI 금지 상태에 있는지를 지시하고, IN_SMM_INDICATION은 SMM 지시자 맵이다. 비트와이즈 OR의 결과, ACCOUNTED는 비트맵, 예를 들어 SMRAM에 저장될 수 있다. 이 분석 후에, 제어는 ACCOUNTED 상태가 모든 존재 스레드에 대해 활성인지가 판단될 수 있는 다이아몬드 450으로 이동된다(다이아몬드 450). 이것은 ACCOUNTED 동작의 결과와 존재 맵 사이의 비교에 기초하여 결정될 수 있다. 그렇지 않으면, 제어는 블록 440으로 다시 이동된다. 그렇지 않으면, 제어는 블록 455로 이동되며, 여기서 SMI 이벤트가 처리될 수 있다. 따라서, 모나크 스레드는 소망의 SMM 코드를 수행할 수 있다. 모나크 스레드에 의해 수행되는 SMM의 결론에 있어서, 제어는 블록 460으로 이동된다. 블록 460에서, ACCOUNTED 상태 및 SMM 지시자 맵가 재설정될 수 있다(블록 460). 즉, 모나크 스레드는 이 비트 맵의 모두에 값을 재설정할 수 있다. 그 다음 모나크 스레드는 그것이 SMI로부터 재개될 수 있는 다른 논리 프로세서에 신호를 보낼 수 있다(블록 465). 이와 같이, 다른 스레드는 그 대기 루프로부터 방출된다. 따라서, 블록 475에서 모든 스레드는 SMM으로부터 개재될 수 있다. 도 4의 실시예에서 이러한 특정 구현이 도시되어 있을지라도, 본 발명의 범위는 이와 관련하여 한정되지 않는다.

[0037] 따라서, 실시예는 메모리 종속성을 갖지 않고 SMM 핸들러 실행을 가능하게 해서, 신뢰성을 개선한다. 또한, 이 메커니즘은 SMI 핸들링이 멀티 코어/멀티 소켓 시스템에서 병목이 되는 것을 회피할 수 있도록 SMM과 관련된 성능 및 확장성 문제를 처리한다. 따라서, 실시예는 DRAM 종속성을 가지고 SMM 코드의 실행을 회피해서, SMM 코드가 메모리 에러를 진단 및 정정하는 높은 가용성 사용 모델을 가능하게 한다.

[0038] 실시예는 긴 플로우 또는 SMI 금지 상태에 있는 논리 프로세서의 존재로 레이턴시가 감소된 SMM으로의 진입을 또한 가능하게 한다. 대조적으로, 현재 SMM 코드가 1개 이상의 프로세서가 SMM에 늦게 조인하거나 SMM 금지 상태에 있게 하는지를 판정할 수 있는 어떤 신뢰성 매커니즘이 존재하지 않으므로, 최대 긴 플로우 상태보다 더 큰 타임아웃이 설정된다. 이 해결방안은, 신뢰성이 없고 구현하기에 어려운 것에 더하여, SMM 레이턴시를 증가시키고 OS 실시간 응답을 감소시키고 본 발명의 실시예를 사용하여 극복될 수 있다.

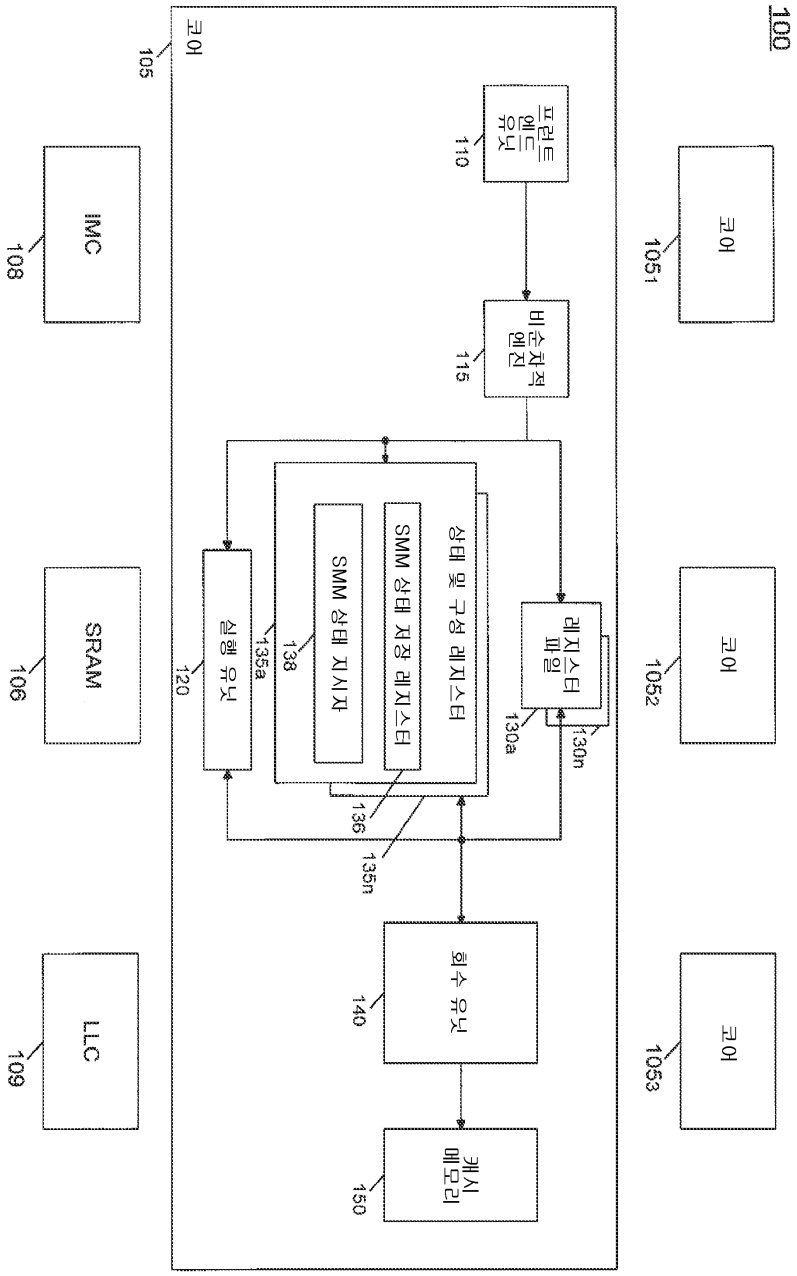
[0039] 실시예는 코드로 구현될 수 있고 명령을 수행하는 시스템을 프로그램하는데 사용될 수 있는 명령을 저장한 저장

매체 상에 저장될 수 있다. 저장 매체는 플로피 디스크, 광 디스크, 고체 상태 드라이드(SSD), 컴팩트 디스크 읽기 전용 메모리(CD-ROM), 컴팩트 디스크 리라이터블(CD-RW), 및 자기 광 디스크를 포함하는 어떤 종류의 디스크, 읽기 전용 메모리(ROM), 동적 랜덤 액세스 메모리(DRAM), 정적 랜덤 액세스 메모리(SRAM)와 같은 랜덤 액세스 메모리(RAM), 소거가능 프로그래가능 읽기 전용 메모리(EPROM), 플래시 메모리, 전기적 소거가능 프로그램가능 읽기 전용 메모리(EEPROM)와 같은 반도체 장치, 자기 또는 광학 카드, 또는 전자 명령을 저장하는데 적당한 다른 종류의 매체를 포함할 수 있지만, 이들에 한정되지 않는다.

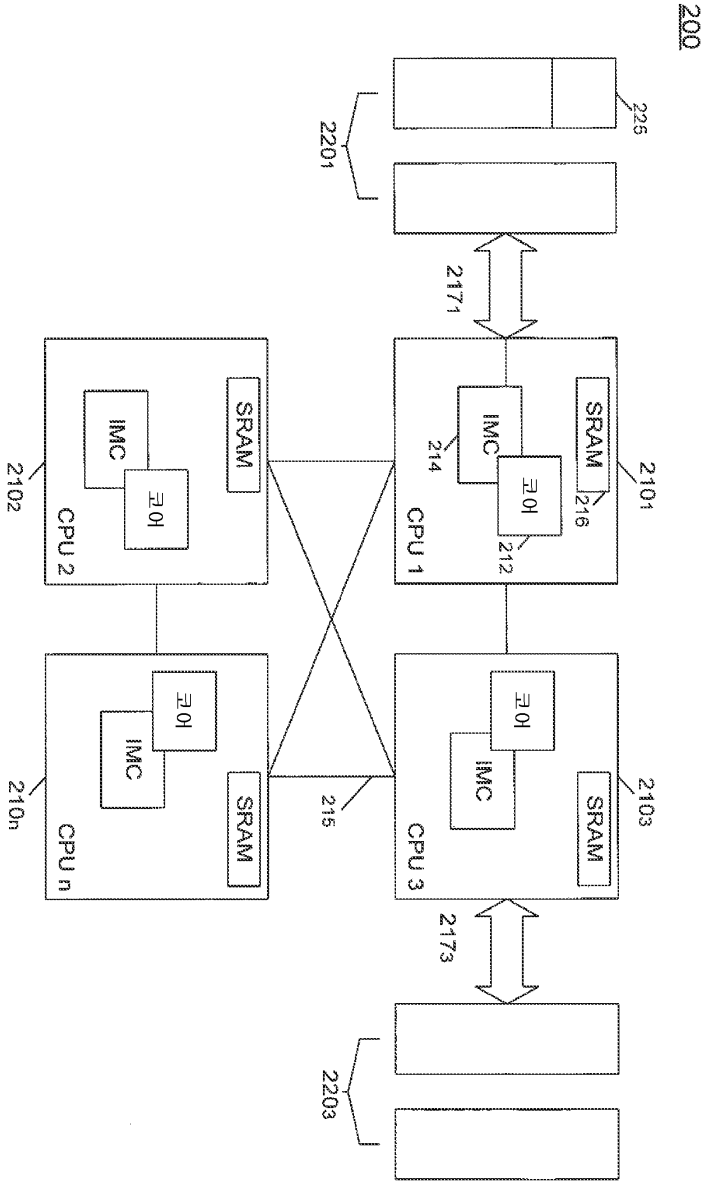
[0040] 본 발명이 한정된 수의 실시예에 대하여 기재되었을지라도, 당업자는 그것으로부터 다수의 수정 및 변화를 인식할 것이다. 첨부된 청구범위는 본 발명의 진정한 정신 및 범위 내에 포함되는 바와 같이 모든 그러한 수정 및 변화를 커버하는 것으로 의도된다.

도면

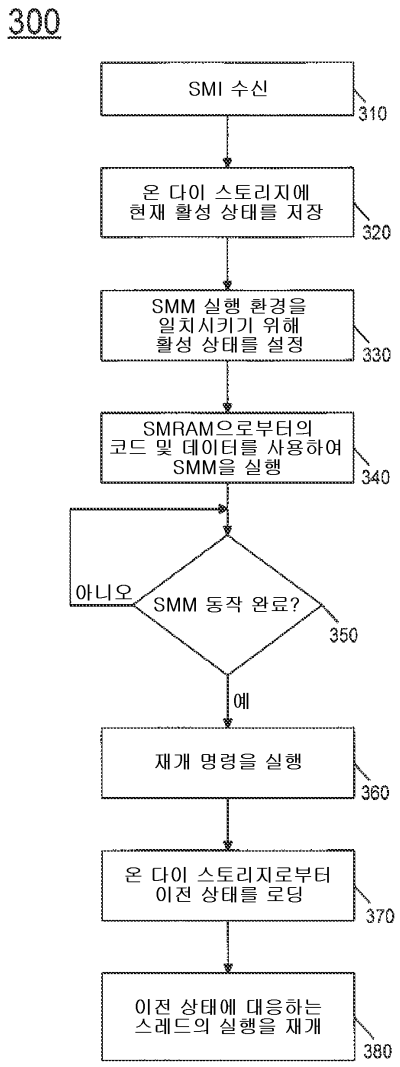
도면1



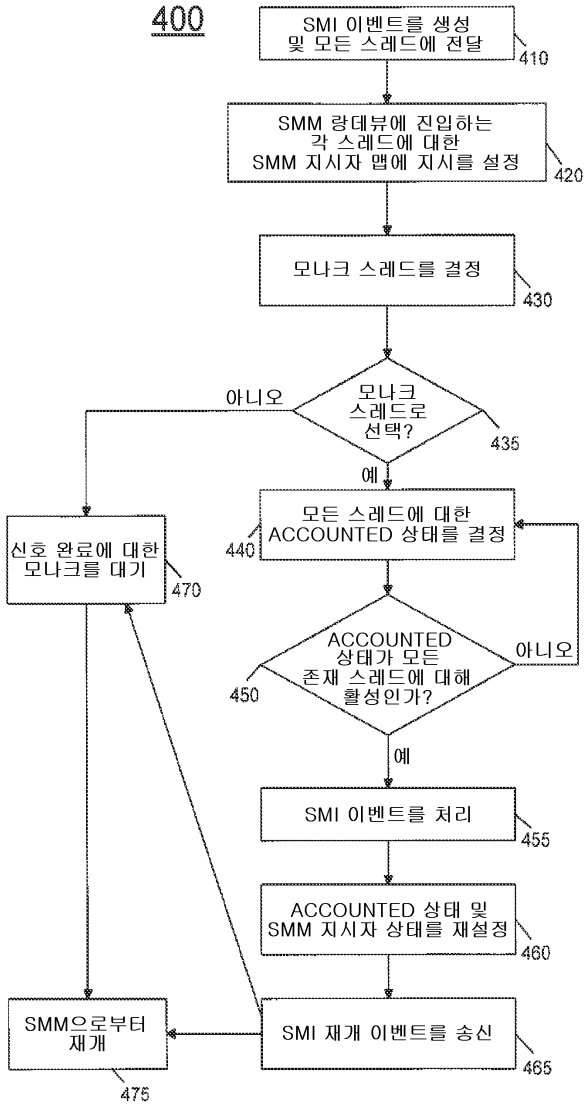
도면2


200
225
2201
2171
CPU 1
IMC
214
코어
212
SRAM
216
2101
CPU 2
IMC
코어
SRAM
2102
CPU n
코어
IMC
SRAM
210n
215
CPU 3
코어
IMC
SRAM
2103
2173
2203

도면3


300
SMI 수신
310
온 다이 스토리지에
현재 활성 상태를 저장
320
SMM 실행 환경을
일치시키기 위해
활성 상태를 설정
330
SMRAM으로부터의
코드 및 데이터를 사용하여
SMM을 실행
340
아니오
SMM 동작 완료?
350
예
재개 명령을 실행
360
온 다이 스토리지로부터
이전 상태를 로딩
370
이전 상태에 대응하는
스레드의 실행을 재개
380

도면4


400
SMI 이벤트를 생성
및 모든 스레드에 전달
410
SMM 랑데뷰에 진입하는
각 스레드에 대한
SMM 지시자 맵에 지시를 설정
420
모나크 스레드를 결정
430
모나크
스레드로
선택?
435
아니오
예
모든 스레드에 대한
ACCOUNTED 상태를 결정
440
ACCOUNTED
상태가 모든
존재 스레드에 대해
활성인가?
450
아니오
예
SMI 이벤트를 처리
455
ACCOUNTED 상태 및
SMM 지시자 상태를 재설정
460
SMI 재개 이벤트를 송신
465
신호 완료에 대한
모나크를 대기
470
SMM으로부터
재개
475