(19)대한민국특허청(KR)
(12) 등록특허공보(B1)

(51) 。Int. Cl.
G06F 9/50 (2006.01)

(45) 공고일자 2006년07월13일
(11) 등록번호 10-0600925
(24) 등록일자 2006년07월06일

(21) 출원번호 10-2003-0091764
(22) 출원일자 2003년12월16일

(65) 공개번호 10-2004-0065981
(43) 공개일자 2004년07월23일

(30) 우선권주장 10/346,147 2003년01월16일 미국(US)

(73) 특허권자 인터내셔널 비지네스 머신즈 코포레이션
미국 10504 뉴욕주 아몬크 노쓰 캐슬 드라이브

(72) 발명자 헤이르멘,피터,조셉
미국미네소타55906,로체스터,오크뷰드라이브엔이3012

허플러,칼,로버트
미국미네소타55955-7048,멘토빌,262애비뉴60752

메이,헨리,조셉
미국미네소타55955-0066,멘토빌,디어리지코트308

보센,케네스,찰스
미국미네소타55955,멘토빌,225애비뉴60823

(74) 대리인 신영무
이용미

심사관 : 이재근

(54) 스레드 유형에 근거한 컴퓨터 자원의 동적 할당

요약

스레드에 연관된 특정"유형"에 근거하여 복수의 물리적 부시스템들을 포함하는 멀티스레드(multithreaded) 컴퓨터의 컴퓨터 자원들에 스레드를 동적으로 할당하는 방법 및 기록매체가 개시된다. 구체적으로, 컴퓨터내의 동일한 물리적 부시스템들내에 존재하는 자원들에 스레드 유형이 할당되어, 특정 스레드 유형의 신규 생성된 스레드 및/또는 재활성 스레드들이 그들 각각의 스레드 유형에 할당된 자원에 동적으로 할당되도록 해준다. 이와 같이, 동일한 유형을 공유하는 스레드들은 컴퓨터의 동일한 물리적 부시스템내에 존재하는 컴퓨터 자원들에 일반적으로 할당되고, 이는 컴퓨터내에 존재하는 다수의 물리적 부시스템들간의 상호 트래픽을 줄여주며, 이에 따라 전체적인 시스템 성능이 개선된다.

대표도

도 3

색인어

멀티스레드, 컴퓨터 자원, 동적 할당, 스레드 유형

명세서

도면의 간단한 설명

도 1은 본 발명에 따른 동적 자원 할당을 통합시킨 멀티스레드(multithreaded) 컴퓨터의 주요 하드웨어 구성요소 블록도이다.

도 2는 도 1의 멀티스레드 컴퓨터의 주요 소프트웨어 구성요소 블록도이다.

도 3은 도 2에서 언급된 스레드 디스패처(thread dispatcher)에 의해 실행되는 스레드 활성 루틴의 프로그램 흐름을 도시한 흐름도이다.

도 4는 도 2에서 언급된 스레드 디스패처(thread dispatcher)에 의해 실행되는 또다른 스레드 활성 루틴의 프로그램 흐름을 도시한 흐름도이다.

도 5는 도 2에서 언급되고 도 4의 활성 스레드 루팅과 관련하여 이용되는 운영체제에 의해 실행되는 어플리케이션/논리적 부시스템 개시 루틴의 프로그램 흐름도이다.

도 6a 내지 6d는 본 발명에 부합하는 방식으로 멀티스레드 컴퓨터의 2개의 물리적 부시스템간에 컴퓨터 자원을 할당하는 예를 설명하는 블록도이다.

도 7은 본 발명에 부합하는 방식으로 멀티스레드 컴퓨터의 2개의 물리적 부시스템간에 컴퓨터 자원을 할당하는 또다른 예를 설명하고, 특정 스레드 유형에 하나 이상의 물리적 부시스템의 컴퓨터 자원들을 할당하는 것을 설명하는 블록도이다.

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 컴퓨터 및 컴퓨터 소프트웨어에 관한 것으로, 더 구체적으로, 멀티스레드 컴퓨터에서 컴퓨터 자원을 할당하는 것에 관한 것이다.

현대 사회에서 컴퓨터에 대한 신뢰도가 계속 증가함에 따라, 컴퓨터 기술은 증가하는 요구에 맞추기 위해 많은 면에서 앞서가야 한다. 중요한 연구개발 노력의 특정된 주제중 하나는 병렬화(parallelism), 즉, 병렬로 다수의 타스크를 수행하는 것에 관한 것이다.

증가된 병렬 처리를 용이하게 하기 위해 다수의 컴퓨터 소프트웨어 및 하드웨어 기술들이 개발되었다. 소프트웨어 관점에서, 멀티스레드(multithreaded) 운영 체제 및 커널(kernel)들이 개발되었는데, 이는 컴퓨터 프로그램들이 다수의 "스레드(thread")들로 동시에 실행되어 다수의 타스크들이 실질적으로 동시에 수행될 수 있도록 해준다. 스레드는 일반적으로 프로그램 실행의 독립적 경로를 나타낸다. 예를 들어, 전자상거래 컴퓨터 어플리케이션의 경우에, 상이한 스레드들이 상이한 고객들에게 할당되어 각 고객의 특정 전자상거래 트랜잭션이 별도의 스레드로 처리되도록 해준다.

하드웨어 관점에서, 컴퓨터들은 증가된 워크로드 능력을 제공하기 위해 복수의 마이크로프로세서들에 점차적으로 의지한다. 더욱이, 일부 마이크로프로세서들은 복수의 스레드들을 병렬로 실행할 수 있는 능력을 지원하도록 개발되었으며, 복수의 마이크로프로세서를 이용함으로써 달성할 수 있는 것과 동일한 다수의 성능 이득을 제공한다.

그러나, 멀티-프로세서 컴퓨터에서 발생할 수 있는 심각한 병목 현상은 각 마이크로프로세서간의 데이터 전송과 관련되어 있는데, 이를 통상 통신 손실(communication cost)이라 지칭한다. 대부분의 컴퓨터들은 컴퓨터의 주요 작업 저장장치로

서의 역할을 하는 주메모리에 의존한다. 그러나, 주메모리로부터 데이터를 검색하고 주메모리에 데이터를 재저장하는 것은 데이터가 마이크로프로세서내에서 내부적으로 전송되는 속도보다 훨씬 느린 속도로 수행되도록 요구된다. 데이터가 마이크로프로세서에 의해 이용될 때 주메모리로부터의 데이터를 일시 저장하기 위해, 캐쉬라 불리우는 중개 버퍼가 종종 활용된다. 이러한 캐쉬들은 메모리에 비해 크기가 작고 속도가 빠르다. 캐쉬는 데이터의 시공간적 국부성(temporal and spatial locality)의 장점을 취하고, 그 결과, 컴퓨터에서 발생하는 상대적으로 느린 주메모리 액세스 회수를 줄이고 컴퓨터에 의해 드는 전체적 통신 손실을 감소시켜준다.

일반적으로, 컴퓨터내의 모든 마이크로프로세서들이 동일한 메모리를 공유하게 되는데, 이러한 구조를 대칭적 멀티프로세싱(Symmetric Multiprocesing: SMP)이라 지칭한다. 그러나, 마이크로프로세서와 주메모리간의 모든 통신이 공통의 버스 또는 상호접속선을 통해 이루어진다는 전형적인 요구사항으로 인해 컴퓨터는 한계에 직면한다. 컴퓨터내의 마이크로프로세서 개수가 증가함에 따라, 주메모리에 대한 통신 트래픽은 중개 캐쉬의 이용여부에 상관없이 시스템 성능상의 장애가 된다.

이러한 잠재적인 장애를 해결하기 위해, 다수의 컴퓨터 설계는 비균일 메모리 액세스(Non-Uniform Memory Access: NUMA)에 의존하고, 여기서 다수의 주메모리들이 컴퓨터에 걸쳐 필연적으로 분산되고 마이크로프로세서 및 캐쉬 집합들과 물리적 부시스템(subsytem) 또는 모듈로 물리적으로 그룹화된다. NUMA 컴퓨터의 각 물리적 부시스템내의 마이크로프로세서, 캐쉬 및 메모리는 전형적으로 동일한 회로보드 또는 카드에 장착되어 물리적 부시스템에 "지역적(local)"인 모든 구성요소들간에 상당히 고속의 상호동작을 제공한다. 물리적 부시스템은 또한 시스템 버스 또는 점대점 상호접속선 집합과 같은 네트워크를 통해 서로간에 결합되어, 하나의 물리적 부시스템내의 마이크로프로세서가 또다른 물리적 부시스템내에 저장된 데이터를 액세스할 수 있도록 해줌으로써, 컴퓨터의 전체 성능을 효과적으로 확장시킨다. 그러나, 메모리 액세스는 국부 메모리(즉, 마이크로프로세서와 동일한 물리적 부시스템내에 존재하는 메모리)에 저장된 데이터의 액세스 시간이 원격 메모리(즉, 또다른 물리적 부시스템에 존재하는 메모리)에 저장된 데이터의 액세스시간에 비해 상당히 짧다는 점에서 "비-균일(non-uniform)"이라 언급된다.

따라서, 통신 손실 관점에서 보면, 각 물리적 부시스템내의 데이터 트래픽을 국부화시키고 물리적 부시스템들간에 전달될 필요가 있는 데이터의 수를 줄임으로써 NUMA 컴퓨터에서 성능은 최대화된다.

컴퓨터에서 하드웨어자원의 효율적인 이용은 일반적으로 소프트웨어와 하드웨어간의 협력을 요한다. 상기에서 언급한 바와 같이, 소프트웨어 관점에서, 컴퓨터에 의해 수행되는 작업의 대부분은 다양한 스레드들에 의해 처리된다. 최적의 성능을 보장하기 위해, 일반적으로 스레드들은 컴퓨터의 작업량이 가용 컴퓨터 자원들간에 균등하게 분산되는 방식으로 가용 컴퓨터 자원의 부집합에 할당된다.

예를 들어, 마이크로프로세서의 효율적 이용을 위해, 가용 마이크로프로세서들간에 스레드들을 균등하게 분산하여 각각의 개별 마이크로프로세서들의 작업량의 균형을 맞추는 것이 바람직한데, 이러한 프로세스는 "균형적(symmetric)"자원 할당이라 언급된다. 그러나, 마찬가지로 통신 손실이 시스템 성능상에 중요한 영향을 미칠 수 있다고 한다면, 스레드를 그가 이용할 데이터와 논리적으로 묶음으로써, 해당 스레드에 의한 데이터 액세스가, 가능하면 캐쉬내에서, 또는, NUMA 컴퓨터의 경우에는 적어도 동일한 물리적 부시스템내에서 국부화되도록 하는 것이 또한 바람직하다. 그렇지 않으면, 비-국부화된 데이터의 통신 손실이 스레드의 대칭 분산 이득을 초과할 수 있다. 일반적으로, 데이터와 스레드를 묶는 것은 동일한 유형의 스레드들을 물리적으로 국부화된 메모리, 프로세서들 및 연관된 자원들에 연관시키기 위한 인간의 판단을 필요로 한다.

균형적 자원 관리 방식에서, 스레드는 활성 시점에, 예를 들면, 스레드가 생성되거나 재활성될 때 분산된다. 활성화된 스레드는 가장 많이 이용가능하거나, 적게 로드된 자원 또는 자원 집합에 할당되는 것이 통상적이다. 그러나, 일반적으로, 통신 손실을 해결하기 위한 메모리 자원과 같은 자원의 비-균일 분산은 일반적으로 자동화되고 투명한(transparent) 방식으로 구현되지 않는다. 비-균일 자원 관리는 주로 실질적인 사용자 분석 및 개별 구성(예를 들어, 자원 할당 문제를 구체적으로 다루기 위한 컴퓨터 프로그램들의 맞춤 프로그래밍을 포함함)을 필요로 한다.

발명이 이루고자 하는 기술적 과제

자원 관리는 컴퓨터의 운영체제 또는 커널 레벨에서 보다 바람직하게 다루어지며, 컴퓨터상에 설치될 수 있는 어플리케이션 또는 기타 컴퓨터 프로그램들에 적용되는 특정의 프로그래밍 기법과는 무관하다. 특히, 자원 관리가 운영체제 또는 커널에서 구현될 때, 이는 컴퓨터 자원의 최적 할당을 지원하기 위해 상위 레벨의 컴퓨터 프로그램들의 특정의 맞춤화를 요구하지 않고, 따라서 소정의 컴퓨터에서 실행되는 장차 모든 컴퓨터 프로그램들에 성능 이득을 제공한다. 특히, 개개의 물

리적 부시스템내에서 스레드-활용 자원간의 국부화(localization)를 통해 성능상의 이득이 달성되는 NUMA 컴퓨터에서, 특별한 맞춤화(significant customization)를 필요로 하지 않고, 좀더 투명한 방식으로 효율적인 자원 할당을 구현하는 것은 매우 바람직할 것이다.

발명의 구성 및 작용

본 발명은 스레드에 연관된 특정 "유형(type)"에 근거하여 컴퓨터 자원에 스레드가 동적으로 할당되는 장치, 프로그램 제품 및 방법을 제공함으로써 종래 기술과 연관된 전술한 문제점 및 기타 문제점들을 해결한다. 구체적으로, 컴퓨터내에서 동일한 물리적 부시스템내에 존재하는 자원들에 스레드 유형이 할당되어서, 특정 스레드 유형의 새롭게 생성되고/되거나 재활성된 스레드들이 각각의 스레드 유형에 할당된 자원들에 동적으로 지정된다. 이와 같이, 동일한 유형을 공유하는 스레드들은, 실질적으로 투명한 방식으로, 이러한 스레드들이 연관된 컴퓨터 프로그램들의 심각한 맞춤화를 필요치 않고도, 컴퓨터의 동일한 물리적 부시스템내에 존재하는 컴퓨터 자원들에 일반적으로 지정된다.

이하에서 좀더 명백하게 되는 바와 같이, 스레드들을 다양한 스레드 유형으로 분류하기 위해, 임의 개수의 스레드 속성 또는 특성들이 활용될 수 있다. 기타 가능한 구별중에서, 스레드 유형들은 실행 우선순위, 버퍼에 대한 지정, 사용자 신원(identity), 사용자 프로파일, 메모리 부시스템, 부모 타스크, 부모 스레드, 부모 잡(parent job), 부모 어플리케이션 및 사용자 권한중 하나 이상에 근거하여 정의될 수 있다.

대부분의 예에서, 본명세서에서 설명되는 스레드 할당은 특정 물리적 시스템내의 자원이 다른 물리적 부시스템내에 존재하는 다른 자원들과 상호작용할 필요성을 상당히 줄일 것이다. 대신에, 특정 스레드에 대한 통신 트래픽 및 처리 오버헤드는 하나의 물리적 부시스템내로 국한될 가능성이 높아져서, 부시스템간 통신에 관련된 오버헤드를 줄이고, 시스템 성능을 최대화시킨다.

일부 실시예에서, 주어진 스레드 유형과 연관된 자원은 단일의 물리적 부시스템내로 국한될 수 있다. 그러나, 다른 실시예에서, 스레드 유형은 다수의 물리적 부시스템의 자원에 할당될 수 있다. 그러나, 후자 실시예에서도, 소정 스레드 유형의 개별 스레드들을 그 유형에 대해 자원들이 할당되어 있는 물리적 부시스템들중 하나에 존재하는 자원들에만 할당하는 것이 여전히 바람직할 것이다.

본 발명을 특징짓는 전술한 장점들 및 기타 장점들은 본명세서에 첨부되고 본 명세서의 한 부분을 이루는 청구범위에 기재되어 있다. 그러나, 본 발명 및 이에 따른 장점 및 목적들의 더 나은 이해를 위해, 도면 및 발명의 상세한 설명에 기재된 관련 설명을 참조하여야 할 것이다.

본 명세서에서 설명되는 실시예들은 다수의 물리적 부시스템을 포함하는 유형의 멀티스레드 컴퓨터에서 적어도 부분적으로는 스레드 유형에 근거하여 스레드들에 자원을 할당하는 동적 자원 메커니즘을 활용한다. 본 발명에 따르면, 특정의 스레드 유형이 특정의 자원 집합에 연관될 때마다 상기 스레드 유형에 일치하고 활성되는 모든 후속 스레드들은 동일한 자원 집합에 할당될 것이다. 자원 집합은 전형적으로 컴퓨터 내의 복수의 물리적 부시스템들간의 상호 트래픽(cross traffic)을 최소화시키기 위해 개개의 물리적 부시스템에 국한되어, 전체적인 시스템 성능을 최적화한다.

이와 같이, 본 발명에 따른 실시예에서, 자원의 균형적 할당은 전형적으로 유일한 유형의 스레드가 생성될 때에만 발생한다. 그렇지 않고, 자원이 이미 할당되어 있는 다른 스레드들과 동일한 스레드 유형을 공유하는 스레드에 대해서는 불균형(asymmetric) 할당이 발생한다. 다른 비균일 자원 할당과 유사하게, 유형별로 스레드를 그룹화하는 것은 특정 스레드에 필요한 자원들이 심각한 지연없이 손쉽게 이용가능할 확률을 증가시킨다는 측면에서 장점을 제공한다.

이하에서 설명될 실시예에서, 일반적으로 자원 할당은 컴퓨터 시스템내에서 비교적 투명하게 구현될 수 있다. 스레드를 유형화하기 위한 규칙들이 일단 정의되면, 자원 할당은 명시적인 사용자 관리없이 통상적으로 발생할 수 있다. 이는 종래에 명시된 구성, 맞춤화 및 사용자 분석을 필요로 하는 종래의 NUMA 구조와는 대조를 이룬다. 또한, 이는 복수의 스레드들간의 특정 유사성과는 무관하게 스레드들이 생성되거나 재활성되는 시점에서 가장 이용가능한 자원들에 스레드가 할당되는 종래의 대칭적 자원 관리 방식과도 대조를 이룬다.

본 명세서의 설명은 특정 자원 집합에 할당되는 실체(entities)를 "스레드(threads)"라 언급할 것이다. 그러나, 컴퓨터 시스템의 유일한 실행 경로를 정의하는 실체를 설명하기 위해 다른 용어들도 이용될 수 있다. 이와 같이, 용어 "스레드"는 컴퓨터 시스템에서 특정 실행 경로를 정의하는 컴퓨터 내의 임의의 실체에 대응하는 것으로 간주되어야 할 것이다.

본 발명과 관련된 스레드 유형은 실제적으로 임의의 스레드 속성 또는 기타 구별될 수 있는 특성을 포함할 수 있으며, 예를 들어, 실행 우선순위, 동일한 가상 또는 물리적 버퍼 또는 메모리로의 할당, 사용자 신원, 부모 논리 부시스템(parent logical subsystem), 잡(job), 어플리케이션, 타스크 또는 스레드, 동일한 메모리 부시스템으로의 할당, 스레드가 개시될 때 실행될 초기 프로그램의 명칭, 스레드 권한(authority) 및 사용자 프로파일이 포함되며, 여기에 한정되는 것은 아니다.

또한, 자원 집합은, 프로세서, 로컬 캐쉬, 공유 캐쉬, 공유 메모리 등과 같은 컴퓨터 자원의 임의의 조합을 포함할 수 있다. 또한, 컴퓨터 자원은 다양한 입/출력 자원들과 같은 다른 유형의 자원들도 포함할 수 있다. 전형적으로, 불균형의 친근성(asymmetric affinities)을 통해 집합으로서 할당하기에 적합한 자원들이 컴퓨터의 별개의 특정 물리적 부시스템내에 위치하는데, 여기서 물리적 부시스템은 다른 물리적 부시스템내의 자원들과 상호작용할 때보다 서로간에 좀더 효율적으로 상호작용하는 컴퓨터 자원그룹으로 통상 간주된다. 이하에서 설명될 실시예에서, 예를 들어, 물리적 부시스템은 동일한 모듈에 배치된, 예를 들면, 동일한 회로 또는 멀티칩 모듈(multi-chip module: MCM)상에 위치하거나, 이에 의해 직접 액세스되거나, 이에 의해 제어되는 하드웨어 자원들을 그룹화함으로써 정의된다. 예를 들면, 인터내셔널 비즈니스 머신즈(IBM)의 eServer iSeries 중간급 컴퓨터 시스템에서, 물리적 부시스템에는 다수의 프로세서와 공유 메모리와 함께 다양한 레벨(예, L1, L2 및/또는 L3)의 중간 공유 및/또는 로컬 캐쉬를 포함하는 특정의 별개 MCM들이 포함된다. 더욱이, 일부 예에서, 메모리는 물리적 부시스템의 나머지들과 분리된 카드상에 배치될 수 있지만, 그럼에도불구하고 MCM에 존재하는 제어기를 이용하여 직접 액세스될 수 있다. 이러한 실시예에서, 소정의 MCM상에 존재하는 자원들간의 통신은 상이한 MCM들에 존재하는 자원들간의 통신보다 통상적으로 훨씬 빠르다. 이와 같이, 유사한 스레드들을 단일의 MCM상에 배치된 자원들에 할당함으로써, MCM간의 통신 트래픽은 MCM 내부간의 통신 증가에 의해 최소화되고, 따라서 컴퓨터의 전체 성능이 최적화될 수 있다.

예시된 실시예에서, 스레드들이 활성될 때마다(예, (그들이 신규인 경우에) 생성될 때마다 및/또는 (존재하지만 현재 비활성이거나 수면상태인 경우에) 재활성될 때마다) 특정 자원 집합에 할당된다. 그러나, 특정 스레드 유형에 자원 집합을 할당하는 것은 상이한 실시예에서 상이한 시점에 적절하게 수행될 수 있다. 예를 들어, 일실시예에서, 스레드 유형에 자원을 할당하는 것은 아직 어떠한 자원도 할당되지 않은 유형 및/또는 부가적인 자원이 필요한 유형의 스레드 활성과 관련하여 수행될 수 있다. 그러나, 다른 실시예에서, 스레드 유형에 자원을 할당하는 것은 해당 유형의 스레드 활성화 이전에, 예를 들면, 어플리케이션의 개시, 논리적 부시스템의 개시와 관련하여, 또는 특정 스레드 유형에 자원을 미리 할당하라는 특정 프로그램 명령어에 응답하여 수행될 수 있다. 이와 관련하여, 논리적 부시스템은 실질적으로 서로가 논리적으로 관련된 어플리케이션, 잡, 스레드 또는 타스크들의 임의의 수집물을 포함할 수 있다.

또한, 이하에서 더욱 명백해지는 바와 같이, 일부 실시예에서 스레드 유형은 컴퓨터 시스템 내의 복수의 물리적 부시스템과 연관된 컴퓨터 자원 집합에 할당될 수 있다. 예를 들면, 특정의 스레드 유형이 소정의 물리적 부시스템내에서 이용가능한 자원들 이상을 필요로 할 것으로 예측되는 경우에 복수의 물리적 부시스템으로부터 컴퓨터 자원들을 할당하는 것이 바람직할 수 있다. 또한, 하나의 물리적 부시스템에 배치된 자원들이 충분히 이용되지 못하고 있는 경우에, 복수의 물리적 부시스템들이 로드를 공유하도록 허용하는 것이 바람직할 수 있다. 그러나, 이러한 예에서, 세부 유형(subtype)을 정의하는 것, 또는 적어도 소정 유형의 특정 스레드들을 하나의 물리적 시스템에만 위치하는 자원 집합에 할당하는 것이 바람직할 수 있다 (예를 들면, 소정 스레드의 경우에, 단일 MCM상에 위치하는 프로세서 및 메모리에 이 스레드를 할당한다).

다른 실시예에서, 소정의 물리적 부시스템에 이용가능한 컴퓨터 자원들의 부집합만을 포함하는 컴퓨터 자원 집합에 스레드 유형이 할당될 수 있다.

이제, 도면들(전체 도면에 걸쳐서 유사한 도면부호는 유사한 부분을 지칭함)을 참조하면, 도 1은 본 발명에 부합하는 동적 자원 할당을 통합시킨 컴퓨터(10)의 주요 하드웨어 구성요소들을 도시한다. 컴퓨터(10)는, 예를 들어, 네트워크 서버, 중간급 컴퓨터, 메인프레임 컴퓨터(예, AS/400, eServer iSeries 중간급 컴퓨터)등과 같은 임의 수의 복수-사용자 컴퓨터를 통상적으로 나타낸다. 그러나, 본 발명은, 예를 들어, 워크스테이션, 데스크톱 컴퓨터, 휴대용 컴퓨터 등과 같은 기타 컴퓨터 및 데이터 처리 시스템 또는 다른 프로그래머블 전자 장치(예, 내장형 제어기 등을 포함함)내에도 구현될 수 있음이 이해될 것이다.

컴퓨터(10)는 통상적으로 시스템 버스 또는 기타 통신 인터페이스를 통해 서로 연결된 복수의 물리적 부시스템(12)들을 포함한다. 또한, 컴퓨터(10)는 저장장치, 워크스테이션, 단말, 네트워크, 이미지 장치 등과 같은 다양한 유형의 자원들을 포함하는 다양한 I/O 자원(16)을 통상적으로 포함한다.

각각의 물리적 부시스템(12)은, 예를 들어, 공유 메모리(18), 하나 이상의 마이크로프로세서(20) 및 공유캐쉬(22) 및 하나 이상의 로컬 캐쉬(24)를 포함하는 하나 이상 레벨의 캐쉬 메모리와 같은 컴퓨터 자원 집합을 포함한다. 각 물리적 부시스

템(12)의 자원들(18-24)은 다른 물리적 부시스템들(12)에 있는 자원과 상호작용 또는 통신하는 것보다 서로간에 더 효율적으로 상호작용 또는 통신하는 능력을 갖는 것으로 특징지워진다. 예를 들어, 각 물리적 부시스템(12)내의 자원들은 동일한 멀티-칩 모듈(MCM) 또는 회로 보드에 배치될 수 있고, 이에 의해 이러한 자원들간의 상호연결선은 시스템 버스(14)를 통해 연결되는 상호연결선보다 10배 또는 그 이상 빠를 수 있다.

컴퓨터 자원들간의 다른 방식의 물리적 분할이 본 발명의 다른 실시예에서 이용될 수 있다. 더욱이, 다른 유형의 컴퓨터 하드웨어 구조들이 본 명세서에 기재된 동적 자원 할당 기법을 활용할 수 있다. 따라서, 본 발명은 도 1에 도시된 특정의 하드웨어 구조에 제한되는 것은 아니다.

도 2는 도 1의 컴퓨터(10)에 의해 활용될 수 있는 예시적인 컴퓨터 구조(30)를 도시한다. 도시된 바와 같이, 구조(30)는 다수의 잡 또는 어플리케이션이 그 위에서 실행되는 운영 체제(32)에 의존할 수 있다. 일부 실시예에서, 하나 이상의 어플리케이션들이 공통의 논리적 부시스템(36)내에서 서로 연관되어 있으며, 다른 어플리케이션들은 임의의 특정 논리적 부시스템에 연관되어 있지 않을 수 있다.

도 2에 도시된 바와 같이, 멀티스레드 컴퓨터가 되는 컴퓨터(10)는 사용자들을 위해 요청된 타스크들을 수행하기 위해 다수의 스레드들(38)을 실행하거나 처리할 수 있다. 스레드(38)는 운영체제(32)내에, 특정 어플리케이션(34) 내에, 특정 논리적 부시스템(36) 내에, 및/또는 구조(30)의 어딘가를 포함하는, 다수의 배경(context)에서 활용될 수 있다. 임의의 구조에서, 복수의 논리적 할당은 균형있게 정의될 수 있으며, 이에 의해, 독립적으로 실행되는 운영체제를 포함하는 복수의 논리적 분할이 소정의 구조내에서 발견될 수 있다. 따라서, 스레드(38)는 실질적으로 소정의 컴퓨터 구조내의 어디에선가 논리적으로 존재할 수 있다.

특정 스레드에 자원을 할당하는 것을 포함하는 스레드(38)의 관리는 스레드 디스패처(thread dispatcher, 40)에 의해 통상 수행되는데, 일반적으로 스레드 디스패처(40)는 운영체제(32)내에 존재한다. 전술한 eServer iSeries 구현에서, 예를 들면, 스레드 디스패처(40)는 이 컴퓨터의 라이센스 내부 코드(Licensed Internal Code: LIC)내에 존재할 수 있다. 또한, 본 발명에 따른 일부 실시예에서 스레드 디스패처는 컴퓨터내의 가능한 스레드 부집합만을 관리할 수 있음을 이해할 것이다.

일반적으로, 본 발명의 실시예를 구현하기 위해 실행되는 루틴들은, 운영체제의 일부로서 또는 특정 어플리케이션, 컴포넌트(component), 프로그램, 객체, 모듈 또는 명령 시퀀스, 또는 이들의 부집합으로서 구현되든 간에, 본 명세서에서는 "컴퓨터 프로그램 코드" 또는 단순히 "프로그램 코드"라고 지칭될 것이다. 프로그램 코드는 전형적으로 컴퓨터 내의 다양한 메모리 및 저장장치에 다양한 시점에 존재하는 하나 이상의 명령어들을 통상적으로 포함하고, 컴퓨터내의 하나 이상의 프로세서들에 의해 판독되고 실행될 때, 본 발명의 다양한 특징을 구현하는 단계 또는 요소들을 실행하기 위해 필요한 단계들을 컴퓨터가 수행하도록 만들 것이다. 또한, 본 발명은 완전히 컴퓨터 또는 컴퓨터 시스템을 기능시키것에 관련하여 설명될 것이지만, 본 기술분야의 통상의 지식을 가진 자는 본 발명의 다양한 실시예들이 다양한 유형의 프로그램 제품으로서 배포될 수 있고, 본 발명은 이러한 배포를 실제 수행하는데 사용된 신호전달매체의 특정 종류에 상관없이 동일하게 적용될 수 있음을 이해할 것이다. 신호전달매체의 예에는 휘발성 및 비휘발성 메모리 장치, 플로피 및 기타 착탈식 디스크, 하드 디스크 드라이브, 자기 테이프, 광디스크(예, CD-ROM, DVD 등)같은 기록형 매체와, 디지털 및 아날로그 통신 링크와 같은 전송형 매체가 포함되며, 이에 국한되는 것은 아니다.

또한, 본 명세서에서 기재되는 다양한 프로그램 코드는 그것이 본 발명의 특정 실시예로 구현되는 어플리케이션 또는 소프트웨어 컴포넌트에 기초하여 식별될 수 있다. 그러나, 이후에 나오는 임의의 특정 프로그램 명명법은 단지 편의상 이용되는 것이며, 본 발명이 이러한 명명법에 의해 식별되고/되거나 암시되는 임의의 특정 어플리케이션에만 사용되는 것으로 한정되어서는 안 될 것이다. 더욱이, 컴퓨터 프로그램들이 루틴, 프로시져, 방법, 모듈, 객체 등으로 구성되는 수많은 방식들 뿐만 아니라, 프로그램 기능이 전형적인 컴퓨터내에 존재하는 다양한 유형의 소프트웨어 계층(예, 운영체제, 라이브러리(libraries), API, 어플리케이션, 애플릿 등)들에 할당되는 다양한 방식에 따라, 본 발명은 본 명세서에 기재된 프로그램 기능의 지정된 구성 및 할당에 국한되는 것은 아님을 이해하여야 할 것이다.

본 기술분야의 숙련자들은 도 1 내지 2에 도시된 예시적인 환경이 본 발명을 제한하려는 것은 아님을 인식할 것이다. 본 기술분야의 당업자들은 다른 대안의 하드웨어 및/또는 소프트웨어 환경들이 본 발명의 범위을 벗어남없이 사용될 수 있음을 인식할 것이다.

이제, 본 발명의 구체적인 실시예를 참조하면, 도 3은 스레드를 활성시키기(예, 생성 또는 재활성시키기) 위한 요청에 응답하여, 도 2의 디스패처(40)에 의해 실행될 수 있는 스레드 활성화 루틴(50)을 도시한다. 루틴(50)은 블록(52)에서 스레드에 연관된 스레드 유형을 판단함으로써 시작한다. 전술한 바와 같이, 스레드를 유형별로 구분하기 위해 임의의 스레드 특

성 또는 속성들이 이용될 수 있다. 예를 들어, 이하에서 설명되는 예는 부모 어플리케이션 또는 논리적 부시스템에 의해 정의된 스레드 유형에 중점을 두어서, 특정 어플리케이션/논리적 부시스템을 대신하여 개시되는 모든 스레드는 동일한 유형을 공유할 것이다. 일반적으로, 해당 유형의 스레드가 상이한 자원 집합보다는 공통의 자원 집합에 할당되는 경우에 상대적으로 나은 성능을 낳게 되는 임의의 스레드 특성 또는 속성이 본 발명에 부합하도록 스레드를 분류하기 위해 이용될 수 있다.

스레드 유형이 판단되면, 블록(54)은 판단된 스레드 유형이 이미 자원집합에 할당되어 있는지 여부를 판단한다. 먼저, 판단된 스레드 유형에 자원이 할당되어 있지 않다고 가정하면, 블록(54)은 (예, 하나의 물리적 부시스템 또는 공지된 물리적 부시스템 집합에 국부화된 가장 이용가능한 자원 집합에 균형적으로 할당함으로써) 상기 판단된 스레드 유형에 물리적 부시스템상의 자원 집합을 할당하는 블록(56)으로 제어를 전달한다. 예를 들어, 상이한 자원 집합이 상이한 물리적 부시스템과 연관된 경우에, 블록(56)은 가장 이용가능한 물리적 부시스템의 자원을 상기 판단된 스레드 유형에 지정할 수 있다. 또한, 하나의 물리적 부시스템상에 존재하는 자원들을 할당하는 것이 가능하지 않거나 실용적이지 않다면, 복수의 물리적 부시스템상의 구분되는 자원 집합들을 하나의 유형에 연관시키기 위해 스레드 유형이 복수의 세부 유형들로 투명성있게 세분될 수 있다.

자원들이 스레드 유형에 할당되면, 블록(58)은 판단된 스레드 유형을 위해 할당된 자원들에 활성될 스레드를 지정한다. 다음에, 스레드는 종래의 방식으로 활성되고, 루틴(50)은 종료한다.

블록(54)으로 돌아와서, 상기 판단된 스레드 유형에 대해 자원들이 이미 할당되어 있다면, 블록(56)은 그냥 통과되고, 제어는 새로운 스레드에 그 스레드 유형에 대해 이미 할당된 자원들을 지정해주는 블록(58)으로 바로 진행한다. 대안적으로, 도 3에 도시된 바와 같이, 블록(54)은 해당 스레드 유형에 대해 추가적으로 자원이 필요한지를 동적으로 판단하는 블록(60)으로 제어를 전달할 수 있다. 예를 들어, 특정 스레드 유형에 연관된 자원들이 완전히 활용되었다면, (가능하면, 동일한 물리적 부시스템내에 배치된 자원들을 통상적으로 이용하여) 그 스레드 유형에 추가적인 자원을 할당하는 블록(56)으로 제어를 전달하는 것이 바람직할 수 있다. 그렇지 않으면, 제어는 블록(60)에서 블록(58)으로 진행한다. 그러나, 다른 실시예에서, 추가적인 자원의 동적 할당은 지원되지 않을 수 있다.

도 3의 루틴(50)을 이용하여, 아직 어떠한 자원도 미리 할당되지 않은 유형을 갖는 스레드의 활성과 관련하여 자원들이 스레드 유형에 대해 할당되는 것이 보여진다. 그러나, 도 3의 루틴에 대한 대안으로서, 스레드 활성화가 별도의 동작으로 소정의 스레드 유형에 대해 컴퓨터 자원을 미리 할당하는 것이 바람직할 수 있다. 예를 들어, eServer iSeries 컴퓨터에서, 사용자 잡이 개시될 때 물리적 부시스템내의 메모리 풀(pool)에 사용자 잡이 지정되는 것이 전형적이다. 메모리 풀은 일반적으로 논리적 부시스템이 개시되거나 시작될 때 논리적 부시스템에 할당된다. 따라서, 다수 그룹의 메모리 및 프로세서들이 컴퓨터 내에서 이용가능한 실시예에서(예, 물리적 부시스템들이 복수의 MCM상에 정의되어 있을 때), 하나의 MCM 또는 물리적 부시스템상의 지정된 메모리 풀에 대하여 가능할때마다, 그리고, 논리적 부시스템이 개시 또는 시작되는 시점에서 모든 메모리를 할당하고자 하는 것이 바람직할 수 있다. 더욱이, 풀이 너무 큰 경우에는 서브풀(sub-pool)이 정의되어 개별 MCM상에 위치할 수 있으며, 통상적으로 사용자에게는 알려지지 않는 방식으로 이루어진다. 본 발명에 부합되게, 스레드 디스패처는 연관 부시스템 또는 메모리 풀에서 실행되는 잡에게 해당 부시스템 또는 메모리 풀에 연관된 프로세서들에게 높은 친근성을 부여하도록 구성되는 것이 예측된다.

도 4 및 5는 부모 어플리케이션 또는 논리적 부시스템의 시작과 관련하여 스레드 유형에 대한 자원 할당이 수행되는 대안적인 실시예를 예시적으로 설명한다. 도 4에 도시된 바와 같이, 이 실시예에서 활성 스레드 루틴(70)은, 도 3의 블록(52)에서 수행되는 프로세스와 유사하게, 블록(72)의 스레드 유형을 판단함으로써 실행될 수 있다. 스레드 유형이 판단되면, 상기 판단된 스레드 유형에 대해 자원이 이미 할당되어 있는 것으로 가정된다. 이러한 경우에, 제어는 해당 유형에 미리 할당되어 있던 자원을 신규 스레드에 지정하는 블록(74)으로 바로 진행한다. 루틴(70)은 그런 후에 종료될 것이다. 또한, 이러한 대안적인 실시예에서도, 추가 자원의 필요성에 대한 검사가 필요하면 수행될 수 있다.

도 5에 도시된 바와 같이, 어플리케이션/논리적 부시스템의 개시 루틴(80)은 어플리케이션 또는 논리적 부시스템을 시작하라는 요청이 수신될 때마다 실행될 수 있다. 루틴(80)은 시작되는 어플리케이션 또는 논리적 부시스템에 연관된 스레드 유형을 생성함으로써 블록(82)에서 시작된다. 다음에, 블록(84)은, 예를 들어, 균형적 할당을 이용하고, 하나의 물리적 부시스템에 배치된 자원을 통상적으로 이용하여, 신규-생성된 스레드 유형을 위해 물리적 부시스템상의 자원들을 할당한다. 전술한 바와 같이, 하나의 물리적 부시스템상에 존재하는 자원들을 할당하는 것이 가능하지 않거나 실용적이지 않은 경우에, 스레드 유형은 복수의 물리적 부시스템상에 있는 구별되는 자원 집합들과 유형을 연관시키기 위해 복수의 세부 유형으로 투명하게 세분될 수 있다.

다음, 블록(86)은 요청된 어플리케이션 또는 논리적 부시스템을 개시시키고, 요구되는 스레드를 활성시키고, 루틴(80)이 종료한다. 블록(86)에서 활성된 임의의 스레드에 대하여, 도 4에 관련하여 전술한 스레드를 활성시키기 위해 루틴(70)이 호출될 것이다.

본 발명을 구현하는 한 방식의 실제적 예는 도 6a 내지 6d를 참조하여 이하에서 설명될 것이다. 컴퓨터가 2개의 물리적 부시스템들(예, 프로세서 카드 또는 MCM)을 포함하고 있고, 각각은 복수의 프로세서 및 다수의 메모리를 포함하고 잇는 것으로 가정한다. 각 물리적 부시스템의 메모리는 모든 프로세서들에게 액세스될 수 있지만, 메모리 액세스는 정보가 요청 프로세서와 동일한 물리적 부시스템으로부터 검색되는 경우에 가장 빠르다. 따라서, 각각의 프로세서에 대한 메모리 액세스의 대부분이 국부화될 수 있다면 유익할 것이다. 이러한 경우에, 자원 그룹은 연관된 프로세서, 캐쉬 및 메모리를 갖춘 카드이다. 2개의 물리적 부시스템이 도 6a에 PS A 및 PS B로 도시되어 있으며, 각각에 대한 자원 그룹은 프로세서(P), 캐쉬(C) 및 메모리(M) 자원들에 의해 나타내어진다.

이제 도 6b를 참조하여, 50여개의 스레드를 실행시킬 제1의 수취계정(Accounts Receivable) 어플리케이션을 지원하기 위해 논리적 부시스템이 개시된다고 가정한다. 50개의 모든 스레드가 유사한 유형으로 분류되기 때문에, 본 발명에 부합하게, 이러한 스레드들 모두 제1의 물리적 부시스템 PS A에 할당될 것이다. 도 6b의 PSA에 자원에 적용된 빗금 표시에 의해 나타내어지는 바와 같이, 통상적으로 물리적 부시스템의 모든 프로세서 및 캐쉬들이 수취계정 어플리케이션의 스레드들에 의해 사용되기 위해 이용가능할 것이다. 더욱이, 물리적 부시스템의 가용 메모리의 부집합을 나타내는 메모리 풀도 이러한 스레드에 의해 사용되기에 이용가능하다.

그러나, 일부 실시예에서, 수취계정 어플리케이션의 모든 스레드들이 전적으로 제1의 물리적 부시스템에만 항상 지정될 수 없을 수도 있음을 이해하여야 할 것이다. 특히, 예를 들어, 수취계정 어플리케이션이 시스템상에서 실행되는 유일한 어플리케이션이라면, 그의 지정된 자원을 넘어서 스레드 또는 어플리케이션의 활용을 확장시키는 것이 바람직한 상황이 존재할 수 있음을 이해하여야 할 것이다. 스레드와 그들의 자원간의 "친근성(affinity)"의 개념은 할당 규칙이 항상 고정되어 있는 것이 아니라, 특정 환경이 보장된다면 시간에 따라 변할 수 있음을 종종 나타낸다.

이제, 도 6c를 참조하면, 제2의 물품목록 제어(Inventory Control) 어플리케이션이 별개의 논리적 부시스템에서 개시된다고 가정한다. 예를 들어 설명하기 위해, 이 어플리케이션의 스레드들은 (별개의 논리적 부시스템에 수취 계정 어플리케이션이 존재한다는 것에 의해) 수취 계정 어플리케이션과 연관된 스레드들과는 구분되는 "유형"을 갖는 것으로 간주한다. 이러한 경우에, 이들 스레드들을 가장 덜 사용되는 자원 집합(이 경우에는, 제2의 물리적 부시스템 PS B)에 할당하기 위해 균형적 할당이 이용될 수 있으며, 이에 따라, 도 6c에 도시된 추가 빗금에 의해 설명되는 바와 같이, 프로세서 및 캐쉬 자원뿐만 아니라 메모리 풀의 할당이 이루어진다.

이제, 도 6d를 참조하면, 제3의, 고객정보 어플리케이션이 개시되고, 이러한 어플리케이션의 유형에 연관된 스레드들이 덜 활동적인 프로세서/메모리 그룹에 할당될 것이라고 가정한다. 예를 들기 위해, 제2의 물리적 부시스템이 가장 덜 활동적이라고 가정한다. 도 6d의 추가적인 빗금에 의해 도시된 바와 같이, 프로세서 및 캐쉬 자원들은 물품목록 제어 및 고객 정보 어플리케이션들에 연관된 스레드 유형들에 사용되기 위해 할당되어 있다. 그러나, 일반적으로, 동일한 물리적 부시스템내에 존재하지만, 구분되는 메모리 풀들이 각 스레드 유형을 위해 유지된다.

전체적인 면에서 자원들이 균등하게 이용되지 않더라도, 각각의 스레드들은 그들의 스레드"유형"에 의해 처리될 데이터에 효과적으로 액세스할 것이기 때문에, 이들은 보다 높은 활용도로써 자원에 대하여 더 효율적으로 동작할 것이다. 하루가 끝나는 시점에서, 수취 계정 어플리케이션의 활동은 크게 줄어들고 입금 어플리케이션이 개시되어, 임금 "유형"의 투명하고 자동적인 할당은 가장 이용가능한 자원으로 갈 것이고, 아마도 이는 수취계정 어플리케이션에 의해 이전에 다량으로 이용되었던 것이다. 그러나, 여분의 수취계정 작업의 상당량이 필요하여 어플리케이션의 활동이 임금 어플리케이션의 개시 이전에 중지되지 않는 경우에, 자동 할당은 운영자 또는 시스템 관리자의 간섭없이 다른 프로세서/메모리 자원 그룹을 선택할 수 있다.

도 7은 다음에 도 6a 내지 6d와 관련하여 논의된 것과 유사한 예를 설명한다. 그러나, 도 7의 예에서, 제1 스레드의 유형에는 양쪽의 물리적 부시스템들로부터의 자원이 할당되는 반면에, 제2의 스레드 유형에는 단지 하나의 물리적 부시스템으로 자원이 할당된다. 예를 들어, 수취계정 어플리케이션은 하나의 물리적 부시스템에 의해 제공될 수 있는 것보다 더 많은 자원을 필요로 하는 반면에, 목록제어 어플리케이션은 비교적 낮은 자원 요구도를 갖는 경우이다. 수취 계정 스레드 유형에는 다수의 물리적 부시스템들로부터 자원이 할당되는 반면에, 해당 유형의 개별 스레드들은 투명하게 "세부 유형"으로 세분될 수 있고, 이는 통상적으로 단지 하나의 물리적 부시스템으로부터의 자원에 할당될 것이며, 따라서, 스레드별로 자원

의 국부성을 유지하고자 한다. 기본 유형의 신규 스레드들은 다양한 물리적 부시스템들의 전체적 가용성에 기반하여 동일한 기본 유형에 지정되는 특정 유형의 세부 유형에 지정될 수 있다. 이러한 가용성은, 예를 들어, 스레드 카운트, 프로세서 활용도 및/또는 메모리 활용도를 포함하는 여러 요인에 따라 기반을 둘 수 있다.

본 명세서의 유익성을 갖는 기타 변경들도 본 기술분야의 당업자들에게 명확하게 이해될 것이다. 따라서, 본 발명은 첨부된 청구범위에 있다.

발명의 효과

본원 발명에 따르면, 동일한 유형을 공유하는 스레드들이, 실질적으로 투명한 방식으로, 이러한 스레드들이 연관된 컴퓨터 프로그램들의 심각한 맞춤화를 필요치 않고도, 컴퓨터의 동일한 물리적 부시스템내에 존재하는 컴퓨터 자원들에 동적으로 지정될 수 있다.

(57) 청구의 범위

청구항 1.

복수의 구별되는 물리적 부시스템들을 포함하는 멀티스레드 컴퓨터에서 컴퓨터 자원을 동적으로 할당하는 방법에 있어서,

(a) 복수의 스레드 유형들의 각각의 스레드 유형에 대하여, 상기 각 스레드 유형을 상기 멀티스레드 컴퓨터내의 공통의 물리적 부시스템내에 물리적으로 위치한 컴퓨터 자원 집합과 연관시키는 단계와,

(b) 스레드를 활성시키라는 요청에 응답하여, 상기 스레드를 상기 스레드의 스레드 유형에 연관된 컴퓨터 자원 집합에 지정하는 단계

를 포함하는 방법.

청구항 2.

제1항에 있어서, 상기 복수의 스레드 유형들 중 제1 스레드 유형에 대해 컴퓨터 자원 집합을 연관시키는 단계는

상기 제1 스레드 유형에 연관된 컴퓨터 자원이 없을 때 상기 제1 스레드 유형을 갖는 스레드를 활성시키라는요청에 응답하여 수행되는 방법.

청구항 3.

제1항에 있어서, 상기 복수의 스레드 유형들 중 제1 스레드 유형에 대해 컴퓨터 자원 집합을 연관시키는 단계는 상기 제1 스레드 유형을 갖는 스레드를 활성시키라는 요청을 수신하기 이전에 수행되는 방법.

청구항 4.

제3항에 있어서, 상기 복수의 스레드 유형들중 제1 스레드 유형에 대해 컴퓨터 집합을 연관시키는 단계는 상기 제1 스레드 유형에 대해 자원을 미리-할당하라는 요청에 응답하여 수행되는 방법.

청구항 5.

제3항에 있어서, 상기 복수의 스레드 유형들중 제1 스레드 유형에 대해 컴퓨터 집합을 연관시키는 단계는 상기 멀티스레드 컴퓨터에서 논리적 부시스템을 개시하라는 요청에 응답하여 수행되는 방법.

청구항 6.

제1항에 있어서, 상기 복수의 스레드 유형은, 실행 우선순위, 버퍼 할당, 사용자 신원(identity), 사용자 프로파일(profile), 메모리 부시스템, 부모 타스크, 부모 스레드, 부모 잡(job), 부모 어플리케이션, 부모 논리적 부시스템, 사용자 권한 및 이들의 조합으로 이루어진 그룹으로부터 선택된 특성에 근거하여 구별되는 방법.

청구항 7.

제1항에 있어서, 각각의 물리적 부시스템은 메모리와 적어도 하나의 프로세서를 포함하는 방법.

청구항 8.

제7항에 있어서, 각각의 물리적 부시스템은 복수의 프로세서들을 포함하고, 상기 메모리의 적어도 한 부분은 상기 복수의 프로세서들에 의해 공유되는 방법.

청구항 9.

제7항에 있어서, 각각의 물리적 부시스템은 캐쉬를 더 포함하는 방법.

청구항 10.

제7항에 있어서, 각각의 물리적 부시스템은 특정의 멀티칩 모듈(multi-chip module: MCM)을 포함하는 방법.

청구항 11.

제1항에 있어서, 상기 복수의 스레드 유형들중 제1 스레드 유형은 상기 복수의 물리적 부시스템들중의 제1 부시스템내에 물리적으로 위치한 제1의 컴퓨터 자원 집합과 연관되어 있고, 상기 방법은,

(a) 상기 멀티스레드 컴퓨터에서 상기 제1 물리적 부시스템과 구별되는 물리적 부시스템내에 물리적으로 위치한 제2 컴퓨터 자원 집합을 상기 제1 스레드 유형과 연관시키는 단계와,

(b) 상기 제1 스레드 유형의 스레드를 활성시키라는 요청에 응답하여, 이러한 스레드를 상기 제1 스레드 유형에 연관된 상기 제1 및 제2 컴퓨터 자원 집합중 하나에 지정하는 단계

를 더 포함하는 방법.

청구항 12.

복수의 구별되는 물리적 부시스템을 포함하는 멀티스레드 컴퓨터에서 컴퓨터 자원을 동적으로 할당하는 방법에 있어서, 상기 방법은,

(a) 복수의 스레드 유형들중 제1 스레드 유형에 대해, 상기 멀티스레드 컴퓨터의 제1 및 제2 물리적 부시스템들내에 각각 물리적으로 위치한 제1 및 제2 컴퓨터 자원 집합에 제1 스레드 유형을 연관시키는 단계와,

(b) 상기 제1 스레드 유형의 제1 스레드를 활성시키라는 요청에 응답하여, 상기 제1 스레드가 상기 제1 스레드의 실행중에 상기 제1 컴퓨터 자원 집합내의 자원들을 활용하도록 상기 제1 스레드를 상기 제1 컴퓨터 자원 집합에 지정하는 단계와,

(c) 상기 제1 스레드 유형의 제2 스레드를 활성시키라는 요청에 응답하여, 상기 제2 스레드가 상기 제2 스레드의 실행중에 상기 제2 컴퓨터 자원 집합내의 자원들을 활용하도록 상기 제2 스레드를 상기 제2 컴퓨터 자원 집합에 지정하는 단계

를 포함하는 방법.

청구항 13.
삭제

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

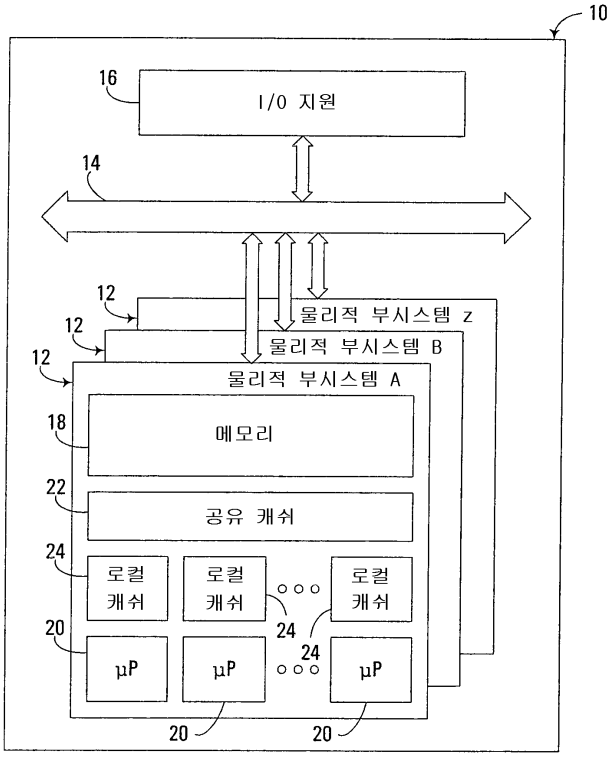
청구항 24.

복수의 구별되는 물리적 부시스템들을 포함하는 유형의 멀티스레드 컴퓨터에 존재하며, 제1항 내지 제11항 중 어느 한 항에 따른 방법을 실행시키기 위해 구성된 프로그램을 기록한 컴퓨터 판독가능한 프로그램 기록매체.

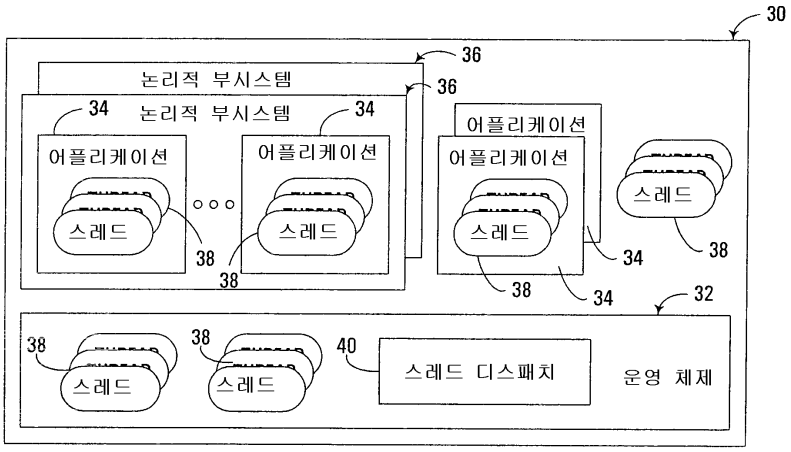
청구항 25.
삭제

도면

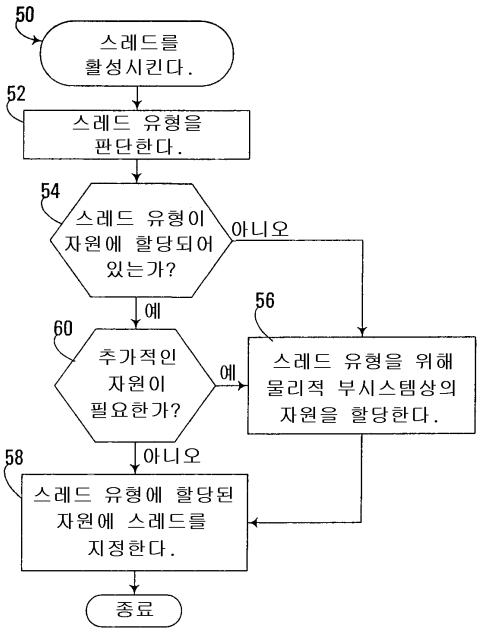
도면1


10
16
I/O 지원
14
12
물리적 부시스템 z
12
물리적 부시스템 B
12
물리적 부시스템 A
18
메모리
22
공유 캐쉬
24
로컬 캐쉬
로컬 캐쉬
로컬 캐쉬
20
μP
μP
μP
24
24
20
20

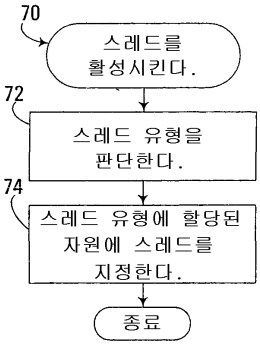
도면2


30
36
논리적 부시스템
36
34
논리적 부시스템
34
어플리케이션
어플리케이션
어플리케이션
어플리케이션
스레드
스레드
스레드
스레드
38
38
38
34
38
34
32
38
38
40
스레드
스레드
스레드 디스패치
운영 체제

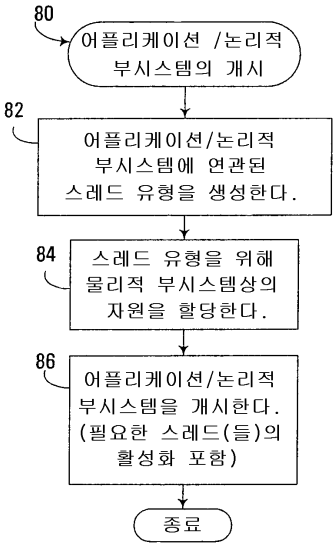
도면3


50
스레드를 활성시킨다.
52
스레드 유형을 판단한다.
54
스레드 유형이 자원에 할당되어 있는가?
아니오
예
56
스레드 유형을 위해 물리적 부시스템상의 자원을 할당한다.
60
추가적인 자원이 필요한가?
예
아니오
58
스레드 유형에 할당된 자원에 스레드를 지정한다.
종료

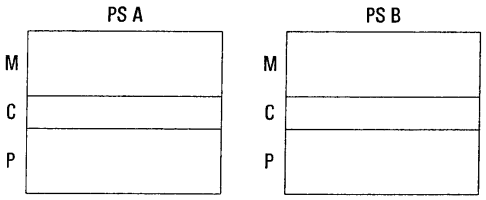
도면4


70
스레드를 활성시킨다.
72
스레드 유형을 판단한다.
74
스레드 유형에 할당된 자원에 스레드를 지정한다.
종료

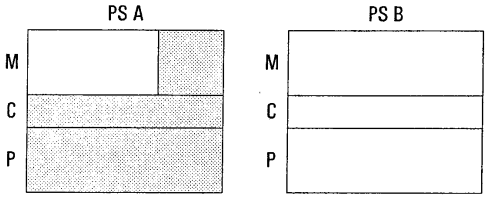
도면5


80
어플리케이션 /논리적 부시스템의 개시
82
어플리케이션/논리적 부시스템에 연관된 스레드 유형을 생성한다.
84
스레드 유형을 위해 물리적 부시스템상의 자원을 할당한다.
86
어플리케이션/논리적 부시스템을 개시한다. (필요한 스레드(들)의 활성화 포함)
종료

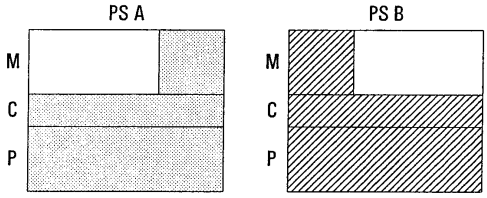
도면6a



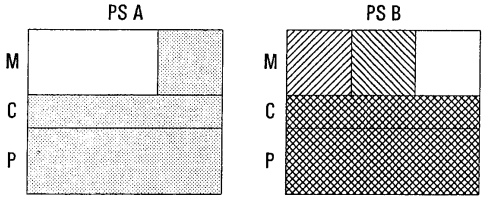
도면6b



도면6c



도면6d



도면7