



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2010-0066405
(43) 공개일자 2010년06월17일

(51) Int. Cl.

G06F 9/06 (2006.01) G06F 9/28 (2006.01)
G06F 21/22 (2006.01) G06F 9/50 (2006.01)

(21) 출원번호 10-2009-0121399

(22) 출원일자 2009년12월08일

심사청구일자 2009년12월08일

(30) 우선권주장

12/330,466 2008년12월08일 미국(US)

(71) 출원인

엔비디아 코퍼레이션

미국 캘리포니아 95050 산타 클라라 산 토마스 익스프레스웨이 2701

(72) 발명자

시머탈, 브래드 더블유.

미국 94110 캘리포니아주 샌프란시스코 콘도 넘버 3 도로레스 스트리트 1049

콕스, 마이클 브라이언

미국 94025 캘리포니아주 멘로 파크 알더 플레이스 1

(74) 대리인

양영준, 백만기

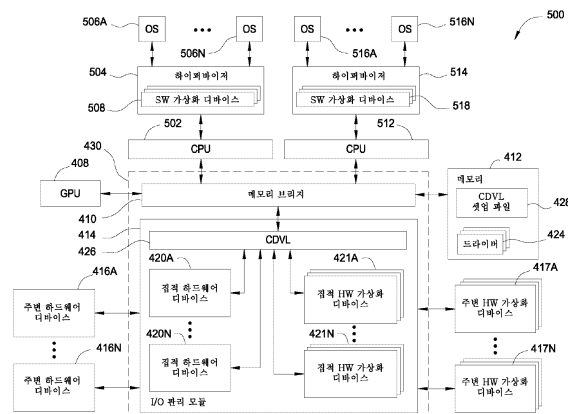
전체 청구항 수 : 총 10 항

(54) 이중 처리 유닛을 위한 중앙집중형 디바이스 가상화 계층

(57) 요약

디바이스들에 대한 액세스를 운영 체제에게 제공하는 방법은, 하드웨어 디바이스들 및 가상화 디바이스들을 열거하는 것을 포함하며, 여기서, 제1 하드웨어 디바이스와 연관된 자원들은 게스트 물리적 자원들로 분할되어 소프트웨어 가상화 디바이스를 생성하고, 제2 하드웨어 디바이스와 연관된 자원들의 다수의 인스턴스들은 애드버타이즈되어 하드웨어 가상화 디바이스를 생성한다. 어느 운영 체제들이 소프트웨어 가상화 디바이스 및 하드웨어 가상화 디바이스에 액세스하도록 허가되는지를 지정하는 제1 및 제2 허가 리스트들이 각각 생성된다. 가상 어드레스 맵들의 제1 및 제2 세트가 생성되며, 여기서, 각각의 세트는 소프트웨어 가상화 디바이스 또는 하드웨어 가상화 디바이스와 연관된 어드레스 공간을 대응하는 허가 리스트에 포함된 각각의 운영 체제와 연관된 어드레스 공간으로 맵핑한다. 본 방법은 허가 리스트들 및 가상 어드레스 맵들에 기초하여 복수의 운영 체제들 각각으로부터의 액세스 요청들을 조정하는 것을 더 포함한다.

대표도



특허청구의 범위

청구항 1

컴퓨팅 디바이스로서,

운영 체제들의 제1 세트를 실행하도록 구성된 제1 프로세서;

제1 하드웨어 디바이스;

상기 제1 하드웨어 디바이스와 연관된 자원들을 게스트 물리적 자원들로 분할하여 소프트웨어(SW) 가상화 디바이스를 생성하도록 구성되며, 상기 제1 프로세서에 의해 실행되는 제1 하이퍼바이저를 저장하는 시스템 메모리;

및

메모리를 포함하는 입/출력(I/O) 관리 모듈을 포함하고,

상기 I/O 관리 모듈에 포함된 상기 메모리는,

상기 운영 체제들의 제1 세트에 포함된 어느 운영 체제들이 상기 제1 하이퍼바이저에 저장된 상기 SW 가상화 디바이스에 액세스하도록 허가되는지를 지정하는 제1 허가 리스트를 생성하는 단계,

각각 상기 제1 하이퍼바이저에 저장된 상기 SW 가상화 디바이스와 연관된 제1 어드레스 공간을 상기 제1 허가 리스트 상에 포함된 상기 운영 체제들 각각과 연관된 어드레스 공간으로 맵핑하는 가상 어드레스 맵들의 제1 세트를 생성하는 단계, 및

상기 제1 허가 리스트에 포함된 상기 운영 체제들이 상기 가상 어드레스 맵들의 제1 세트에 기초해 상기 SW 가상화 디바이스를 이용하여 액세스 동작들을 수행할 수 있게 함으로써, 상기 운영 체제들의 제1 세트에 포함된 각각의 운영 체제로부터의 액세스 요청들을 조정하는 단계

를 수행함으로써 상기 제1 프로세서가 상기 SW 가상화 디바이스에 액세스할 수 있게 하도록 구성된 명령어들을 저장하는

컴퓨팅 디바이스.

청구항 2

제1항에 있어서,

운영 체제들의 제2 세트를 실행하도록 구성된 제2 프로세서; 및

제2 하드웨어 디바이스와 연관된 자원들의 다수의 인스턴스들을 애드버타이즈(advertise)하여 하드웨어(HW) 가상화 디바이스를 생성하도록 구성된 제2 하드웨어 디바이스를 더 포함하고,

상기 시스템 메모리는 상기 제1 하드웨어 디바이스와 연관된 자원들을 게스트 물리적 자원들로 분할하여 제2 SW 가상화 디바이스를 생성하도록 구성되며, 상기 제2 프로세서에 의해 실행되는 제2 하이퍼바이저를 더 저장하고,

상기 I/O 관리 모듈에 포함된 상기 메모리는,

상기 운영 체제들의 제2 세트에 포함된 어느 운영 체제들이 상기 제2 하이퍼바이저에 저장된 상기 제2 SW 가상화 디바이스에 액세스하도록 허가되는지를 지정하는 제2 허가 리스트를 생성하는 단계,

각각 상기 제2 하이퍼바이저에 저장된 상기 제2 SW 가상화 디바이스와 연관된 제2 어드레스 공간을 상기 제2 허가 리스트 상에 포함된 상기 운영 체제들 각각과 연관된 어드레스 공간으로 맵핑하는 가상 어드레스 맵들의 제2 세트를 생성하는 단계, 및

상기 제2 허가 리스트에 포함된 상기 운영 체제들이 상기 가상 어드레스 맵들의 제2 세트에 기초해 상기 제2 SW 가상화 디바이스를 이용하여 액세스 동작들을 수행할 수 있게 하는 단계

를 수행함으로써 상기 제2 프로세서가 상기 제2 SW 가상화 디바이스에 액세스할 수 있게 하도록 구성된 명령어들을 더 포함하는

컴퓨팅 디바이스.

청구항 3

제2항에 있어서,

상기 I/O 관리 모듈에 포함된 상기 메모리는,

상기 운영 체제들의 제1 세트 및 상기 운영 체제들의 제2 세트에 포함된 어느 운영 체제들이 상기 HW 가상화 디바이스에 액세스하도록 허가되는지를 지정하는 제3 허가 리스트를 생성하고,

각각 상기 HW 가상화 디바이스와 연관된 제3 어드레스 공간을 상기 제3 허가 리스트 상에 포함된 상기 운영 체제들 각각과 연관된 어드레스 공간으로 맵핑하는 가상 어드레스 맵들의 제3 세트를 생성하도록 구성된 명령어들을 더 포함하는 컴퓨팅 디바이스.

청구항 4

제3항에 있어서,

상기 I/O 관리 모듈에 포함된 상기 메모리는, 상기 제1 운영 체제가 액세스하고자 하는 제1 하드웨어 디바이스 또는 제1 가상화 디바이스를 지정하는 정보, 및 상기 제1 하드웨어 디바이스 또는 상기 제1 가상화 디바이스와 연관된 하드웨어 디바이스 상에서 상기 제1 운영 체제가 실행하고자 하는 액세스 동작들을 지정하는 정보를 포함하는 상기 운영 체제들의 제1 세트에 포함된 제1 운영 체제로부터 액세스 요청을 수신하도록 구성된 명령어들을 더 포함하는 컴퓨팅 디바이스.

청구항 5

제4항에 있어서,

상기 제1 가상화 디바이스는 상기 SW 가상화 디바이스를 포함하고, 상기 I/O 관리 모듈에 포함된 상기 메모리는,

상기 제1 운영 체제가 상기 제1 허가 리스트에 포함되는 것으로 결정하고,

상기 제1 운영 체제가 상기 SW 가상화 디바이스를 이용하여 상기 액세스 동작들 - 상기 액세스 동작들은 상기 가상 어드레스 맵들의 제1 세트를 이용하여 상기 제1 운영 체제로부터 상기 SW 가상화 디바이스로 라우팅된 구성 동작들 및 MMIO(memory-mapped input/output) 동작들을 포함함 - 을 수행할 수 있게 하도록 구성된 명령어들을 더 포함하는 컴퓨팅 디바이스.

청구항 6

제4항에 있어서,

상기 제1 가상화 디바이스는 상기 HW 가상화 디바이스를 포함하고, 상기 I/O 관리 모듈에 포함된 상기 메모리는,

상기 제1 운영 체제가 상기 제3 허가 리스트에 포함되는 것으로 결정하고,

상기 제1 운영 체제가 상기 HW 가상화 디바이스를 이용하여 상기 액세스 동작들 - 상기 액세스 동작들은 상기 가상 어드레스 맵들의 제3 세트를 이용하여 상기 제1 운영 체제로부터 상기 HW 가상화 디바이스로 라우팅된 구성 동작들 및 MMIO 동작들을 포함함 - 을 수행할 수 있게 하도록 구성된 명령어들을 더 포함하는 컴퓨팅 디바이스.

청구항 7

컴퓨터 시스템에 포함된 하드웨어 디바이스들에 대한 액세스를 운영 체제에 제공하는 방법으로서,

상기 컴퓨터 시스템에 포함된 하나 이상의 하드웨어 디바이스 및 하나 이상의 가상화 디바이스를 열거(enumerate)하는 단계 - 상기 열거하는 단계는, 제1 하드웨어 디바이스와 연관된 자원들이 게스트 물리적 자원들로 분할되어, 소프트웨어(SW) 가상화 디바이스를 생성하는 것으로 결정하고, 제2 하드웨어 디바이스와 연관된 자원들의 다수의 인스턴스들이 상기 제2 하드웨어 디바이스에 의해 애드버타이즈되어 하드웨어(HW) 가상화 디바이스를 생성하는 것으로 결정하는 것을 포함함 - ;

복수의 운영 체제에서의 어느 운영 체제들이 상기 SW 가상화 디바이스에 액세스하도록 허가되는지를 지정하는 제1 허가 리스트를 생성하는 단계;

상기 복수의 운영 체제에서의 어느 운영 체제들이 상기 HW 가상화 디바이스에 액세스하도록 허가되는지를 지정하는 제2 허가 리스트를 생성하는 단계;

각각 상기 SW 가상화 디바이스와 연관된 제1 어드레스 공간을 상기 제1 허가 리스트에 포함된 각각의 운영 체제와 연관된 어드레스 공간으로 맵핑하는 가상 어드레스 맵들의 제1 세트를 생성하는 단계;

각각 상기 HW 가상화 디바이스와 연관된 제2 어드레스 공간을 상기 제2 허가 리스트에 포함된 각각의 운영 체제와 연관된 어드레스 공간으로 맵핑하는 가상 어드레스 맵들의 제2 세트를 생성하는 단계; 및

상기 복수의 운영 체제들 각각으로부터의 액세스 요청들을 조정하는 단계 - 상기 조정하는 단계는, 상기 제1 허가 리스트에 포함된 상기 운영 체제들이 상기 가상 어드레스 맵들의 제1 세트에 기초해 상기 SW 가상화 디바이스를 이용하여 액세스 동작들을 수행할 수 있게 하고, 상기 제2 허가 리스트에 포함된 상기 운영 체제들이 상기 가상 어드레스 맵들의 제2 세트에 기초해 상기 HW 가상화 디바이스를 이용하여 액세스 동작들을 수행할 수 있게 하는 것을 포함함 -

를 포함하는, 하드웨어 디바이스들에 대한 액세스를 운영 체제에 제공하는 방법.

청구항 8

제7항에 있어서,

상기 제1 운영 체제가 액세스하고자 하는 제1 하드웨어 디바이스 또는 제1 가상화 디바이스를 지정하는 정보, 및 상기 제1 하드웨어 디바이스 또는 상기 제1 가상화 디바이스와 연관된 하드웨어 디바이스 상에서 상기 제1 운영 체제가 실행하고자 하는 액세스 동작들을 지정하는 정보를 포함하는 상기 복수의 운영 체제에서의 제1 운영 체제로부터 액세스 요청을 수신하는 단계를 더 포함하는, 하드웨어 디바이스들에 대한 액세스를 운영 체제에 제공하는 방법.

청구항 9

제8항에 있어서,

상기 제1 가상화 디바이스는 상기 SW 가상화 디바이스를 포함하고,

상기 제1 운영 체제가 상기 제1 허가 리스트에 포함되는 것으로 결정하는 단계, 및

상기 제1 운영 체제가 상기 SW 가상화 디바이스를 이용하여 상기 액세스 동작들을 수행할 수 있게 하는 단계를 더 포함하는, 하드웨어 디바이스들에 대한 액세스를 운영 체제에 제공하는 방법.

청구항 10

제8항에 있어서,

상기 제1 가상화 디바이스는 상기 HW 가상화 디바이스를 포함하고,

상기 제1 운영 체제가 상기 제2 허가 리스트에 포함되는 것으로 결정하는 단계, 및

상기 제1 운영 체제가 상기 HW 가상화 디바이스를 이용하여 상기 액세스 동작들을 수행할 수 있게 하는 단계를 더 포함하는, 하드웨어 디바이스들에 대한 액세스를 운영 체제에 제공하는 방법.

명세서

발명의 상세한 설명

기술 분야

[0001] 본 발명은 일반적으로 디바이스 가상화 분야에 관한 것으로, 보다 구체적으로는 이중 처리 유닛들을 위한 중앙 집중형 디바이스 가상화 계층에 관한 것이다.

배경 기술

- [0002] 종래의 컴퓨터 시스템에 있어서, 중앙 처리 유닛(CPU)은 다중 운영 체제(OS)들을 실행할 수 있다. 상이한 OS들의 동작들을 조정하기 위해 "하이퍼바이저(hypervisor)"라고 알려진 소프트웨어 프로그램이 종종 이용된다. 하이퍼바이저는 각각의 OS로부터 컴퓨터 시스템에 포함된 다양한 하드웨어 디바이스들로 전송되는 입력/출력(I/O) 액세스 동작들(액세스들)을 관리한다. 이들 하드웨어 디바이스들은 특히 메모리 모듈들, 컴퓨터 모니터들, 하드 디스크 드라이브들 및 CD-ROM 드라이브들을 포함할 수 있다. 하이퍼바이저는 OS들 각각이, 하드웨어 디바이스의 가상 인스턴스들을 생성하는데 이용될 수 있는 "디바이스 가상화"로 알려진 기술을 이용하여 하드웨어 디바이스에 액세스하는 것을 허가한다. 하드웨어 디바이스는 "소프트웨어 가상화" 및/또는 "하드웨어 가상화"를 통해 가상화될 수 있다.
- [0003] 소프트웨어(SW) 가상화가 구현되는 경우, 하이퍼바이저는 하드웨어 디바이스의 자원들을 다수의 "게스트 물리적 자원들"로 분할한다. 그 다음에 각각의 게스트 물리적 자원은 특정 OS에 할당될 수 있다. 하이퍼바이저는 OS에 게스트 물리적 자원들을 애드버타이즈(advertise)하여, OS가 하드웨어 디바이스와 연관된 게스트 물리적 자원들에 액세스하려고 시도하는 경우 하이퍼바이저가 그 액세스를 트랩하여 해당 OS가 하드웨어 디바이스를 액세스하는 것을 허가하기 전에 올바른 컨텍스트가 하드웨어 디바이스 상에 로딩되는 것을 확실하게 할 수 있도록 한다.
- [0004] 하드웨어(HW) 가상화가 구현되는 경우, 각각의 하드웨어 디바이스는 그 HW 디바이스의 다수의 인스턴스들을 애드버타이즈하여, 각각의 인스턴스에 하드웨어 디바이스에 의해 이용가능하게 만들어진 자원들의 총량이 할당되도록 한다. 그 다음에 각각의 OS에 이들 인스턴스들 중 하나가 할당된다. 하이퍼바이저는 어느 OS가 특정 가상화된 디바이스를 액세스할 수 있는지를 결정함으로써 OS들 간의 조정자로서 동작한다. HW 가상화는 하드웨어에서 구현되기 때문에, 하이퍼바이저는 특정 물리적 자원이 가상화되고 있는 것을 인식하지 못할 수도 있다. 각각의 HW 가상화된 디바이스는 다른 가상화된 디바이스와의 충돌을 인식하고, 물리적 자원 상에서 동작하기 전에 (예를 들면, 해당 디바이스의 하나 이상의 가상화된 인스턴스들을 중지(quiescing)하거나 또는 컨텍스트들을 복원함으로써) 이 충돌들을 해결한다.
- [0005] SW 가상화 및 HW 가상화는 단일 프로세서가 단일 하이퍼바이저의 상부에서 다중 OS를 실행하는 경우 잘 동작한다. 그러나, 다수의 OS들을 실행하는 다중 CPU를 포함하는 더욱 다양한 프로세서 토폴로지들은 SW 가상화 및/또는 HW 가상화에 덜 적당하다. 예를 들면, 멀티-CPU 컴퓨터 시스템은 각각이 하드웨어 디바이스들의 동일한 그룹에 접속된 몇몇의 CPU를 포함할 수 있다. 일부 하드웨어 디바이스들은 하드웨어 또는 소프트웨어로 가상화될 수 없기 때문에, 컴퓨터 시스템은 2개의 CPU가 공유할 수 없는 해당 하드웨어 디바이스에 대해 오직 한개의 자원만을 포함한다. 또한 멀티-CPU 컴퓨터 시스템에서 SW 가상화를 구현하는 것은, 충분히 이용되지 않고 있는 SW 가상화된 하드웨어 디바이스를 OS가 액세스하기를 원하는 경우 병목현상이 발생할 수 있기 때문에 문제가 된다. 하드웨어 디바이스들의 자원들이 게스트 물리적 자원들로 분할되기 때문에, 각각의 OS는 하드웨어 디바이스의 전체 자원 중 오직 일부를 이용할 수 있다. 오직 하나의 OS가 하드웨어 디바이스를 액세스하려고 시도하고 있는 경우에, 그 OS는 게스트 물리적 자원에 의해 할당된 것보다 더욱 많은 자원들을 요구할 수 있다. 하드웨어 디바이스의 자원들이 여전히 이용가능하기 때문에 자원 병목현상이 생기지만, 이 자원들이 다른 게스트 물리적 자원들에 할당되었기 때문에 OS는 이 자원들을 활용할 수 없다.
- [0006] 멀티-CPU 컴퓨터 시스템에서 HW 가상화를 구현하는 것은 또한 가상화된 디바이스가 각각의 CPU 상에서 동작하는 각각의 OS에 대해서 생성되어야 할 필요가 있기 때문에 문제가 된다. 각각의 하드웨어 디바이스에 대하여 HW 가상화를 구현하는 것은 컴퓨터 시스템 내에서 애드버타이즈되는 자원들을 현저히 증가시킬 것이며, 특정 OS가 특정 하드웨어 디바이스에 의해 제공된 자원들을 이용할 필요가 없을지라도 각각의 하드웨어 디바이스가 각각의 OS에 대해 가상화되도록 강요할 것이다.

발명의 내용

해결 하고자하는 과제

- [0007] 따라서, 본 기술 분야에서는 멀티-CPU 컴퓨터 시스템에서 하드웨어 디바이스들을 가상화하기 위한 개선된 기술에 대한 필요성이 남아있다.

과제 해결수단

[0008] 본 발명의 실시예들은 컴퓨터 시스템에 포함된 하드웨어 디바이스들에 운영 체제 액세스를 제공하기 위한 방법을 제공한다. 본 방법은, 상기 컴퓨터 시스템에 포함된 하나 이상의 하드웨어 디바이스들 및 하나 이상의 가상화된 디바이스들을 열거하는(enumerate) 단계 - 상기 열거하는 단계는, 제1 하드웨어 디바이스와 연관된 자원들을 게스트 물리적 자원들로 분할하여 소프트웨어(SW) 가상화된 디바이스를 생성하는 것으로 결정하는 단계, 및 제2 하드웨어 디바이스와 연관된 자원들의 다수의 인스턴스들이 상기 제2 하드웨어 디바이스에 의해 애드버타이즈되어 하드웨어(HW) 가상화된 디바이스를 생성하는 것으로 결정하는 단계를 포함함 -; 복수의 운영 체제 중 어느 운영체제들이 상기 SW 가상화된 디바이스를 액세스하도록 허가될지를 지정하는 제1 허가 리스트를 생성하는 단계; 상기 복수의 운영 체제 중 어느 운영체제들이 상기 HW 가상화된 디바이스를 액세스하도록 허가될지를 지정하는 제2 허가 리스트를 생성하는 단계; 각각의 가상 어드레스 맵이 상기 SW 가상화된 디바이스와 연관된 제1 어드레스 공간을 상기 제1 허가 리스트에 포함된 각각의 운영 체제와 연관된 어드레스 공간으로 맵핑하는 가상 어드레스 맵들의 제1 세트를 생성하는 단계; 각각의 가상 어드레스 맵이 상기 HW 가상화된 디바이스와 연관된 제2 어드레스 공간을 상기 제2 허가 리스트에 포함된 각각의 운영 체제와 연관된 어드레스 공간으로 맵핑하는 제2 세트의 가상 어드레스 맵들을 생성하는 단계; 및 상기 복수의 운영 체제의 각각으로부터의 액세스 요청들을 조정하는 단계 - 상기 조정하는 단계는, 상기 제1 허가 리스트에 포함된 운영 체제들이 상기 제1 세트의 가상 어드레스 맵들에 기초하여 상기 SW 가상화된 디바이스를 이용하여 액세스 동작을 수행하도록 하는 단계, 및 상기 제2 허가 리스트에 포함된 운영 체제들이 상기 제2 세트의 가상 어드레스 맵들에 기초하여 상기 HW 가상화된 디바이스를 이용하여 액세스 동작을 수행하도록 하는 단계를 포함함 - 를 포함한다.

효 과

[0009] 유용하게, 본 발명의 실시예들은 디바이스 가상화를 다수의 CPU를 포함하는 컴퓨팅 시스템들로 확장한다. CDVL은 각각의 CPU 상에서 실행되는 OS가 컴퓨팅 시스템에 연결된 열거된 디바이스들의 공통 풀을 액세스하는 것을 허가한다. CDVL은 SW 가상화 및 HW 가상화와 함께 이용되어 전용 가상화 지원을 요구하는 OS에 그러한 전용 가상화 지원을 제공할 수 있다.

발명의 실시를 위한 구체적인 내용

[0010] 위에서 인용된 본 발명의 특징들이 상세히 이해될 수 있는 방식이 되도록, 위에서 간략히 요약된 본 발명의 보다 특정한 설명이 실시예들을 참조하여 제공될 수 있고, 첨부 도면들에 그 일부가 도시된다. 그러나, 첨부된 도면들은 본 발명의 전형적인 실시예들만을 도시할 뿐, 본 발명의 영역을 제한하는 것으로 고려되지 않으며, 본 발명은 다른 동일하게 유효한 실시예들을 허용할 수 있다.

[0011] 이하의 설명에서, 본 발명에 대한 보다 완전한 이해를 제공하기 위해, 다양한 특정 세부 사항들이 개시된다. 그러나, 당업자라면, 본 발명은 하나 이상의 이들 특정 세부 사항들 없이도 실시될 수 있음을 명백히 알 것이다. 다른 경우, 본 발명을 불명료하게 하지 않도록, 잘 알려진 특징들은 기술되지 않는다.

[0012] 시스템 개요

[0013] 도 1은 본 발명의 하나 이상의 양상들을 구현하도록 구성된 컴퓨터 시스템(100)을 도시하는 블록도이다. 컴퓨터 시스템(100)은 메모리 브리지(105)를 통한 버스 경로를 경유하여 통신하는 중앙 처리 유닛(CPU)(102) 및 시스템 메모리(104)를 포함한다. 도 1에 도시된 바와 같이, 메모리 브리지(105)는 CPU(102) 내에 통합될 수 있다. 대안적으로, 메모리 브리지(105)는 버스를 경유하여 CPU(102)에 접속되는 종래의 디바이스, 예를 들면, 노스브리지 칩(Northbridge chip)일 수 있다. 메모리 브리지(105)는 통신 경로(106)(예를 들면, 하이퍼트랜스포트 링크(HyperTransport link))를 경유하여 I/O(입력/출력) 브리지(107)에 접속된다. 예를 들면, 사우스브리지 칩(Southbridge chip)일 수 있는 I/O 브리지(107)는 하나 이상의 사용자 입력 디바이스들(108)(예를 들면, 키보드, 마우스)로부터의 사용자 입력을 수신하고, 그 입력을 경로(106) 및 메모리 브리지(105)를 경유하여 CPU(102)에 전달한다. 병렬 처리 서브시스템(112)은 버스 또는 다른 통신 경로(113)(예를 들면, PCI Express, AGP(Accelerated Graphics Port), 또는 하이퍼트랜스포트 링크)를 경유하여 메모리 브리지(105)에 연결되며, 일 실시예에서, 병렬 처리 서브시스템(112)은 픽셀들을 디스플레이 디바이스(110)(예를 들면, 종래의 CRT 또는 LCD 기반 모니터)에 전달하는 그래픽 서브시스템이다. 시스템 디스크(114)가 또한 I/O 브리지(107)에 접속된다. 스위치(116)는 I/O 브리지(107)와, 네트워크 어댑터(118) 및 다양한 애드인 카드들(120, 121)과 같은 다른 컴포넌트들 사이의 접속들을 제공한다. USB 또는 다른 포트 접속들, CD 드라이브들, DVD 드라이브들, 녹화 장치들 등을 포함하는 (명시적으로 도시되지 않은) 다른 컴포넌트들이 I/O 브리지(107)에 또한 접속될 수 있다. 도 1에서의 다양한 컴포넌트들을 상호접속하는 통신 경로들은, PCI(Peripheral Component

Interconnect), PCI Express(PCI-E), AGP(Accelerated Graphics Port), 하이퍼트랜스포트(HyperTransport)와 같은 임의의 적절한 프로토콜들, 또는 임의의 다른 버스 또는 점대점 통신 프로토콜(들)을 이용하여 구현될 수 있으며, 상이한 디바이스들 사이의 접속들은 본 기술 분야에 알려진 다른 프로토콜들을 이용할 수 있다.

[0014] 일 실시예에서, 병렬 처리 서브시스템(112)은 예를 들면, 비디오 출력 회로를 포함하는, 그래픽 및 비디오 처리를 위해 최적화된 회로를 통합하며, 그래픽 처리 유닛(GPU)을 구성한다. 다른 실시예에서, 병렬 처리 서브시스템(112)은 본 명세서에서 보다 상세히 기술된 기초 계산 아키텍처를 보존하면서, 범용 처리를 위해 최적화된 회로를 통합한다. 또 다른 실시예에서, 병렬 처리 서브시스템(112)은 메모리 브리지(105), CPU(102) 및 I/O 브리지(107)와 같은 하나 이상의 다른 시스템 요소들과 통합되어, 시스템 온 칩(system on chip; SoC)을 형성할 수 있다.

[0015] 본 명세서에서 도시된 시스템은 예시적인 것이며, 수정 및 변경이 가능함을 이해할 것이다. 브리지들의 수 및 배열을 포함하는 접속 토폴로지는 원하는 대로 변경될 수 있다. 예컨대, 몇몇 실시예에서, 시스템 메모리(104)는 CPU(102)에, 브리지를 통해서가 아니라 직접 접속되며, 다른 디바이스들은 메모리 브리지(105) 및 CPU(102)를 경유하여 시스템 메모리(104)와 통신한다. 다른 대안적인 토폴로지에서는, 병렬 처리 서브시스템(112)은 메모리 브리지(105)에 접속되지 않고, I/O 브리지(107)에 접속되거나 또는 CPU(102)에 직접 접속된다. 또 다른 실시예에서, CPU(102), I/O 브리지(107), 병렬 처리 서브시스템(112) 및 메모리 브리지(105) 중 하나 이상은 하나 이상의 칩에 통합될 수 있다. 본 명세서에서 도시된 특정 컴포넌트들은 선택적인 것이며, 예컨대, 임의의 수의 애드인 카드들 또는 주변 디바이스들이 지원될 수 있다. 몇몇 실시예에서, 스위치(116)는 제거되며, 네트워크 어댑터(118) 및 애드인 카드들(120, 121)은 I/O 브리지(107)에 직접 접속된다.

[0016] 도 2는 본 발명의 일 실시예에 따른, 병렬 처리 서브시스템(112)을 도시한다. 도시된 바와 같이, 병렬 처리 서브시스템(112)은, 각각이 로컬 병렬 처리(PP) 메모리(204)에 접속된 하나 이상의 병렬 처리 유닛들(PPU들)(202)을 포함한다. 일반적으로, 병렬 처리 서브시스템은 U개의 PPU들을 포함하며, 여기서 $U \geq 1$ 이다. (본 명세서에서, 필요한 경우, 유사한 객체들의 다수의 인스턴스들은, 객체를 식별하는 참조 번호들과, 인스턴스를 식별하는 삽입 번호들을 이용하여 표기된다.) PPU들(202) 및 병렬 처리 메모리들(204)은, 프로그램가능한 프로세서들, ASIC들(application specific integrated circuits), 또는 메모리 디바이스들과 같은 하나 이상의 집적 회로 디바이스들을 이용하여 구현되거나, 또는 기술적으로 실현가능성 있는 임의의 그외의 방법으로 구현될 수 있다.

[0017] 도 1을 다시 참조하면, 일부 실시예들에서, 병렬 처리 서브시스템(112)의 PPU들(202)의 일부 또는 전부는, 픽셀 데이터를 저장하고 업데이트하기 위해 (예를 들어, 종래의 프레임 버퍼를 포함하는 그래픽 메모리로서 이용될 수 있는) 로컬 병렬 처리 메모리(204)와 상호작용하고, 픽셀 데이터를 디스플레이 디바이스(110)에 전달하는 등, CPU(102) 및/또는 시스템 메모리(104)에 의해 제공되는 그래픽 데이터로부터 픽셀 데이터를 생성하는 것과 관련된 다양한 작업들을 수행하도록 구성될 수 있는 렌더링 파이프라인들을 갖는 그래픽 프로세서들이다. 일부 실시예들에서, 병렬 처리 서브시스템(112)은 그래픽 프로세서들로서 동작하는 하나 이상의 PPU들(202) 및 범용 연산들에 이용되는 하나 이상의 그외의 PPU들(202)을 포함할 수 있다. PPU들은 동일하거나 상이할 수 있으며, 각각의 PPU는 자신의 전용 병렬 처리 메모리 디바이스(들)을 가질 수 있거나, 전용 병렬 처리 메모리 디바이스(들)을 갖지 않을 수 있다. 하나 이상의 PPU들(202)은 데이터를 디스플레이 디바이스(110)에 출력하거나 각각의 PPU(202)는 데이터를 하나 이상의 디스플레이 디바이스(110)에 출력할 수 있다.

[0018] 동작에서, CPU(102)는 컴퓨터 시스템(100)의 마스터 프로세서이며, 그외의 시스템 컴포넌트들의 동작을 제어하고 조정한다. 특히, CPU(102)는 PPU들(202)의 동작을 제어하는 명령들을 발행한다. 일부 실시예들에서, CPU(102)는 각각의 PPU(202)에 대한 명령들의 스트림을, 시스템 메모리(104), 병렬 처리 메모리(204), 또는 CPU(102) 및 PPU(202) 모두에 의해 액세스가능한 다른 저장 위치에 위치될 수 있는 명령 버퍼(도 1 또는 도 2에 명확하게 도시되지 않음)에 기입한다. PPU(202)는 명령 버퍼로부터 명령 스트림을 판독하고, 그 후 CPU(102)의 동작과 관련하여 비동기적으로 명령들을 실행한다. CPU(102)는 또한 PPU들(202)이 명령 버퍼의 명령들에 응답하여 판독할 수 있는 데이터 버퍼들을 생성할 수 있다. 각각의 명령 및 데이터 버퍼는 PPU들(202) 각각에 의해 판독될 수 있다.

[0019] 다시 도 2를 참조하면, 각각의 PPU(202)는, 메모리 브리지(105)에 (또는, 하나의 대안의 실시예에서, CPU(102)에 직접) 접속되는 통신 경로(113)를 통해 컴퓨터 시스템(100)의 나머지 부분과 통신하는 I/O(입력/출력) 유닛(205)을 포함한다. 컴퓨터 시스템(100)의 나머지 부분으로의 PPU(202)의 접속은 또한 변화될 수 있다. 일부 실시예들에서, 병렬 처리 서브시스템(112)은 컴퓨터 시스템(100)의 확장 슬롯으로 삽입될 수 있는 애드인 카드

로서 구현된다. 그외의 실시예들에서, PPU(202)는 메모리 브리지(105) 또는 I/O 브리지(107)와 같은 버스 브리지를 갖는 단일 칩 상에 집적될 수 있다. 또 다른 실시예들에서, PPU(202)의 일부 또는 전부의 요소는 CPU(102)를 갖는 단일 칩 상에 집적될 수 있다.

[0020] 일 실시예에서, 통신 경로(113)는, 본 기술분야에 주지된 바와 같이, 각각의 PPU(202)에 전용 레인들이 할당되는 PCI-E 링크이다. 그외의 통신 경로들이 또한 이용될 수 있다. I/O 유닛(205)은 전송용 패킷(또는 그외의 신호들)을 통신 경로(113) 상에 생성하고, 또한 모든 인입하는 패킷들(또는 그외의 신호들)을 통신 경로(113)로부터 수신하여, 인입하는 패킷들을 PPU(202)의 적절한 컴포넌트로 보낸다. 예를 들어, 처리 작업들과 관련된 명령들은 호스트 인터페이스(206)로 보내질 수 있는 한편, 메모리 동작들(예를 들어, 병렬 처리 메모리(204)에 대한 판독 또는 그외의 기입)에 대한 명령들은 메모리 크로스바 유닛(210)으로 보내질 수 있다. 호스트 인터페이스(206)는 각각의 명령 버퍼를 판독하고, 명령 버퍼에 의해 지정된 작업을 프런트 엔드(212)로 출력한다.

[0021] 각각의 PPU(202)는 고성능 병렬 처리 아키텍처를 구현한다는 장점을 갖는다. 상세하게 도시된 바와 같이, PPU(202(0))는 C개의 일반 처리 클러스터들(GPC들)(208)을 포함하는 처리 클러스터 어레이(230)를 포함하며, 여기서 $C \geq 1$ 이다. 각각의 GPC(208)는 다수의(예를 들어, 수백 또는 수천개의) 스레드들을 동시에 실행할 수 있으며, 여기서 각각의 스레드는 프로그램의 인스턴스이다. 다양한 어플리케이션들에서, 상이한 GPC들(208)이 상이한 유형들의 프로그램들을 처리하거나 상이한 유형들의 연산들을 처리하기 위해 할당될 수 있다. 예를 들어, 그래픽 어플리케이션에서, GPC들(208)의 제1 세트는 테셀레이션(tessellation) 동작들을 수행하고 패치들(patches)에 대해 프리미티브 토폴로지(primitive topology)를 생성하도록 할당될 수 있고, GPC들(208)의 제2 세트는 프리미티브 토폴로지들에 대한 패치 파라미터들을 평가하는 테셀레이션 셰이딩을 수행하고 정점 위치들(vertex positions) 및 그외의 정점당 속성들(per-vertex attributes)을 결정하도록 할당될 수 있다. GPC들(208)의 할당은 각각의 유형의 프로그램 또는 연산에 대해 발생하는 작업량에 따라 변화할 수 있다. 대안적으로, GPC들(208)은 상이한 처리 작업들 사이에서 전환하는 타임 슬라이스(time-slice) 방식을 이용하여 처리 작업들을 수행하도록 할당될 수 있다.

[0022] GPC들(208)은 실행될 처리 작업들을, 처리 작업들을 정의하는 명령들을 프런트 엔드 유닛(212)로부터 수신하는 작업 분배 유닛(200)을 통해 수신한다. 처리 작업들은 처리될 데이터, 예를 들어, 표면(패치) 데이터, 프리미티브 데이터, 정점 데이터, 및/또는 픽셀 데이터에 대한 포인터들뿐만 아니라 상태 파라미터들 및 데이터가 처리되는 방법(예를 들어, 어느 프로그램이 실행될 것인지)을 정의하는 명령들을 포함한다. 작업 분배 유닛(200)은 처리 작업들에 대응하는 포인터들을 인출하도록 구성될 수 있고, 프런트 엔드(212)로부터 포인터들을 수신할 수 있거나, 프런트 엔드(212)로부터 데이터를 직접 수신할 수 있다. 일부 실시예들에서, 인덱스들은 어레이에서의 데이터의 위치를 지정한다. 프런트 엔드(212)는, 명령 버퍼에 의해 지정된 처리가 개시되기 전에 GPC들(208)이 유효 상태로 구성되는 것을 보장한다.

[0023] 작업 분배 유닛(200)은 처리를 위한 다수의 GPC들(208)로 작업들을 제공할 수 있는 주파수로 작업들을 출력하도록 구성될 수 있다. 본 발명의 일부 실시예에서, GPC들(208)의 부분들은 상이한 유형들의 처리를 수행하도록 구성된다. 예를 들어, 제1 부분은 정점 셰이딩 및 토폴로지 생성을 수행하도록 구성될 수 있고, 제2 부분은 테셀레이션 및 기하학 셰이딩을 수행하도록 구성될 수 있으며, 제3 부분은 렌더링된 화상을 생성하기 위해 스크린에서 픽셀 셰이딩을 수행하도록 구성될 수 있다. 상이한 유형들의 처리 작업들을 효율적으로 수행하기 위해 GPC들(208)의 부분들을 할당하는 능력은, 그러한 상이한 유형들의 처리 작업들에 의해 생성된 데이터의 임의의 확장 및 수축을 수용한다. GPC들(208)에 의해 만들어진 중간 데이터는 버퍼링되어, 다운스트림 GPC(208)에 의해 데이터가 수용되는 레이트가 업스트림 GPC(208)에 의해 데이터가 만들어지는 레이트보다 늦어지는 경우, 중간 데이터가 최소한의 스톨링(stalling)을 갖고 GPC들(208) 사이에 전송되게 할 수 있다.

[0024] 메모리 인터페이스(214)는 각각이 병렬 처리 메모리(204)의 일부분에 접속된 D개의 메모리 파티션 유닛들로 파티션될 수 있으며, 여기서, $D \geq 1$ 이다. 병렬 처리 메모리(204)의 각각의 부분은 일반적으로 하나 이상의 메모리 디바이스들(예를 들어, DRAM(220))을 포함한다. 본 기술분야의 당업자는 DRAM(220)이 그외의 적절한 저장 디바이스들로 대체될 수 있으며, 일반적으로 종래의 설계일 수 있음을 이해할 것이다. 따라서, 상세한 설명은 생략된다. 프레임 버퍼 또는 텍스처 맵들과 같은 렌더 타겟들은 DRAM들(220)에 걸쳐서 저장될 수 있고, 파티션 유닛(215)들이 각각의 렌더 타겟의 부분들을 병렬로 기입하여 병렬 처리 메모리(204)의 이용가능한 대역폭을 효율적으로 이용하게 한다.

[0025] GPC들(208) 중 어느 하나는, 병렬 처리 메모리(204) 내의 파티션 유닛들(215)의 임의의 것에 기입되는 데이터를 처리할 수 있다. 크로스바 유닛(210)은 각각의 GPC(208)의 출력을 임의의 파티션 유닛(214)의 입력 또는 추가

의 처리를 위해 다른 GPC(208)로 보내도록 구성된다. GPC들(208)은 크로스바 유닛(210)을 통해 메모리 인터페이스(214)와 통신하여 다양한 외부 메모리 디바이스들에 대해 관독 또는 기입한다. 일 실시예에서, 크로스바 유닛(210)은 로컬 병렬 처리 메모리(204)에 대한 접속뿐만 아니라 I/O 유닛(205)과 통신하기 위한 메모리 인터페이스(214)에 대한 접속을 가짐으로써, 상이한 GPC들(208) 내의 처리 코어들이 시스템 메모리(104) 또는 PPU(202)에 대하여 로컬이 아닌 그외의 메모리와 통신하게 한다. 크로스바 유닛(210)은 GPC들(208)과 파티션 유닛(215) 간의 트래픽 스트림(traffic stream)을 분리하기 위해 가상 채널들을 이용할 수 있다.

[0026] 다시, GPC들(208)은, 선형 및 비선형 데이터 변환, 비디오 및/또는 오디오 데이터의 필터링, 모델링 동작들(예를 들어, 객체들의 위치, 속도 및 그외의 속성을 결정하기 위해 물리학의 법칙을 적용), 이미지 렌더링 동작들(예를 들어, 테셀레이션 셰이더, 정점 셰이더, 기하학 셰이더, 및/또는 픽셀 셰이더 프로그램들) 등을 포함하지만 이것으로 제한되는 것은 아닌 광범위한 어플리케이션들과 관련된 처리 작업들을 실행하도록 프로그램될 수 있다. PPU들(202)은 데이터를 시스템 메모리(104) 및/또는 로컬 병렬 처리 메모리들(204)로부터 내부(온칩) 메모리로 전달할 수 있고, 데이터를 처리할 수 있고, 결과 데이터를 다시 시스템 메모리(104) 및/또는 로컬 병렬 처리 메모리들(204)에 기입할 수 있으며, 여기서 그러한 데이터는, CPU(102) 또는 다른 병렬 처리 서브시스템(112)을 포함하는, 그외의 시스템 컴포넌트들에 의해 액세스될 수 있다.

[0027] PPU(202)는 로컬 메모리가 제공되지 않는 경우를 포함하여 임의의 양의 로컬 병렬 처리 메모리(204)를 구비할 수 있으며, 로컬 메모리와 시스템 메모리를 임의의 조합으로 이용할 수 있다. 예를 들어, PPU(202)는 통합된 메모리 아키텍처(UMA) 실시예에서의 그래픽 프로세서일 수 있다. 이러한 실시예들에서, 전용 그래픽 (병렬 처리) 메모리가 거의 또는 전혀 제공되지 않을 수 있고, PPU(202)는 시스템 메모리를 배타적으로 또는 거의 배타적으로 사용할 수 있다. UMA 실시예들에서, PPU(202)는 브리지 칩 또는 프로세서 칩 내로 통합될 수 있거나, 브리지 칩 또는 다른 통신 수단들을 통해 시스템 메모리에 PPU(202)를 접속하는 고속 링크(예를 들어, PCI-E)를 갖는 개별 칩으로서 제공될 수 있다.

[0028] 전술한 바와 같이, 임의의 수의 PPU(202)가 병렬 처리 서브시스템(112)에 포함될 수 있다. 예를 들어, 다수의 PPU(202)가 단일 애드인 카드에 제공될 수 있거나, 또는 다수의 애드인 카드가 통신 경로(113)에 접속될 수 있거나, 또는 하나 이상의 PPU(202)가 브리지 칩 내로 통합될 수 있다. 멀티-PPU 시스템 내의 PPU들(202)은 서로 동일하거나 또는 상이할 수 있다. 예를 들어, 상이한 PPU들(202)은 상이한 수의 처리 코어들, 상이한 양의 로컬 병렬 처리 메모리 등을 가질 수 있다. 다수의 PPU(202)가 존재하는 경우, PPU들은 단일 PPU(202)와 가능한 것보다 높은 처리율로 데이터를 처리하도록 병렬로 동작될 수 있다. 하나 이상의 PPU(202)를 포함하는 시스템들은 데스크톱, 랩톱, 또는 핸드헬드 개인용 컴퓨터, 서버, 워크스테이션, 게임 콘솔, 임베디드 시스템 등을 포함하는, 다양한 구성 및 폼 팩터(form factor)로 구현될 수 있다.

[0029] 처리 클러스터 어레이 개요

[0030] 도 3a는 본 발명의 일 실시예에 따른, 도 2의 PPU들(202) 중 하나의 PPU 내의 GPC(208)의 블록도이다. 각각의 GPC(208)는 많은 수의 스레드를 병렬로 실행하도록 구성될 수 있고, 여기서 "스레드"라는 용어는 특정 세트의 입력 데이터에 대해 실행되는 특정 프로그램의 인스턴스(instance)를 가리킨다. 일부 실시예들에서, 복수의 독립적인 명령어 유닛들을 제공하지 않고 많은 수의 스레드의 병렬 실행을 지원하기 위해 SIMD(single-instruction, multiple-data) 명령어 발행 기술들이 이용된다. 다른 실시예들에서, GPC들(208) 중 각각의 GPC 내의 처리 엔진들의 세트에 명령어들을 발행하도록 구성된 공통 명령어 유닛을 이용하여, 많은 수의 일반적으로 동기화된 스레드의 병렬 실행을 지원하기 위해 SIMT(single-instruction, multiple-thread) 기술들이 이용된다. 모든 처리 엔진들이 통상적으로 동일한 명령어들을 실행하는 SIMD 실행 체제와 달리, SIMT 실행은 상이한 스레드들이 주어진 스레드 프로그램을 통해 분기하는 실행 경로들을 더 용이하게 뒤따를 수 있게 한다. 이 기술분야의 통상의 기술자들은, SIMD 처리 체제가 SIMT 처리 체제의 기능적 서브세트를 표현한다는 것을 이해할 것이다.

[0031] 그래픽 어플리케이션들에서, GPC(208)는 프리미티브 설정, 래스터화(rasterization), 및 z 컬링(culling)을 포함할 수 있지만 이에 제한되지 않는 스크린 공간 그래픽 처리 기능들을 수행하기 위해 프리미티브 엔진을 구현하도록 구성될 수 있다. 프리미티브 엔진은 작업 분배 유닛(200)으로부터 처리 작업을 수신하고, 처리 작업이 프리미티브 엔진에 의해 수행되는 동작들을 필요로 하지 않을 때, 처리 작업은 프리미티브 엔진을 통해 파이프라인 관리자(305)에 전달된다. GPC(208)의 동작은 유리하게는 처리 작업들을 스트리밍 멀티프로세서들(SPM들)(310)에 분배하는 파이프라인 관리자(305)를 통해 제어된다. 파이프라인 관리자(305)는 또한 SPM들(310)에 의한 처리된 데이터 출력력을 위한 목적지들을 지정함으로써 작업 분배 크로스바(330)를 제어하도록 구성될 수 있다.

다.

- [0032] 일 실시예에서, 각각의 GPC(208)는 M개의 SPM들(310)을 포함하며, 여기서, $M \geq 1$ 이고, 각각의 SPM(310)은 하나 이상의 스레드 그룹을 처리하도록 구성된다. 또한, 이 기술분야에 알려져 있는 바와 같이, 각각의 SPM(310)은 유리하게는 파이프라인될 수 있는 동일한 세트의 기능 유닛들(예를 들어, 산술 로직 유닛들 등)을 포함함으로써, 이전 명령어가 완료되기 전에 새로운 명령어가 발행될 수 있게 한다. 기능 유닛들의 임의 조합이 제공될 수 있다. 일 실시예에서, 기능 유닛들은 정수 및 부동 소수점 산술(예를 들어, 덧셈 및 곱셈), 비교 연산들, 부울 연산들(AND, OR, XOR), 비트 시프팅, 및 다양한 대수 함수들(예를 들어, 평면 보간법(planar interpolation), 삼각법(trigonometric), 지수, 및 로그 함수들 등)의 계산을 포함하는 다양한 연산들을 지원하며, 동일한 기능 유닛 하드웨어가 상이한 연산들을 수행하는 데 이용될 수 있다.
- [0033] 특정 GPC(208)에 전송된 일련의 명령어들은 본 명세서에서 이전에 정의한 바와 같이, 스레드를 구성하며, SPM(310) 내의 병렬 처리 엔진들(도시되지 않음)에 걸쳐서 동시에 실행하는 스레드들의 특정 수의 모음(collection)을 본 명세서에서는 "스레드 그룹"이라고 부른다. 본 명세서에서 이용되는 바와 같이, "스레드 그룹"은 상이한 입력 데이터에 대해 동일한 프로그램을 동시에 실행하는 스레드들의 그룹을 가리키며, 그 그룹의 각각의 스레드는 SPM(310) 내의 상이한 처리 엔진에 할당된다. 스레드 그룹은 SPM(310) 내의 처리 엔진들의 수보다 적은 스레드들을 포함할 수 있으며, 이 경우, 일부 처리 엔진들은 그 스레드 그룹이 처리되고 있는 사이클들 동안 유휴 상태일 것이다. 스레드 그룹은 또한 SPM(310) 내의 처리 엔진들의 수보다 많은 스레드들을 포함할 수 있으며, 이 경우, 다수의 클록 사이클들에 걸쳐 처리가 일어날 것이다. 각각의 SPM(310)은 최대 G개의 스레드 그룹들을 동시에 지원할 수 있기 때문에, 최대 $G \times M$ 개의 스레드 그룹들이 임의의 주어진 시간에 GPC(208)에서 실행될 수 있다는 것은 자명하다.
- [0034] 부가적으로, 복수의 관련된 스레드 그룹들이 SPM(310) 내에서 동시에 활성화(상이한 실행 페이지 내에서)일 수 있다. 이 스레드 그룹들의 모음을 본원에서는 "CTA(cooperative thread array)"라고 지칭한다. 개개의 CTA의 크기는 $m \times k$ 이고, 여기서 k는 스레드 그룹에서 동시에 실행하는 스레드들의 수이고 전형적으로 SPM(310) 내의 병렬 처리 엔진들의 수의 정수배이며, m은 SPM(310) 내에서 동시에 활성화된 스레드 그룹들의 수이다. CTA의 크기는 일반적으로 CTA에 이용가능한 메모리 또는 레지스터들 등의 하드웨어 자원들의 양과 프로그래머에 의해 결정된다.
- [0035] 배타적인 로컬 어드레스 공간이 각각의 스레드에 대해 이용가능하며, 공유된 퍼-CTA 어드레스 공간이 CTA 내의 스레드들 사이에 데이터를 전달하는 데 이용된다. 퍼-스레드 로컬 어드레스 공간 및 퍼-CTA 어드레스 공간에 저장된 데이터가 L1 캐시(320)에 저장되고, L1 캐시(320)에 데이터를 유지하는 것을 지원하기 위해 추출 정책(eviction policy)이 이용될 수 있다. 각각의 SPM(310)은 로드 및 저장 동작들을 수행하는 데 이용되는 대응하는 L1 캐시(320)의 공간을 이용한다. 각각의 SPM(310)은 또한 스레드들 사이에 데이터를 전달하는 데 이용될 수 있고 모든 GPC들(208) 사이에 공유되는 파티션 유닛들(215) 내의 L2 캐시들에 대한 액세스를 갖는다. 마지막으로, SPM들(310)은 또한 예를 들어 병렬 처리 메모리(204) 및/또는 시스템 메모리(104)를 포함할 수 있는 오프-칩 "전역" 메모리에 대한 액세스를 갖는다. L2 캐시는 전역 메모리에 기입되고 전역 메모리로부터 판독되는 데이터를 저장하는 데 이용될 수 있다. PPU(202)의 외부에 있는 임의의 메모리가 전역 메모리로서 이용될 수 있다는 것을 이해해야 한다.
- [0036] 그래픽 애플리케이션들에서, GPC(208)는 각각의 SPM(310)이 텍스처 맵핑 동작들, 예를 들어, 텍스처 샘플 위치들의 결정, 텍스처 데이터의 판독, 및 텍스처 데이터의 필터링을 수행하기 위해 텍스처 유닛(315)에 연결되도록 구성될 수 있다. 텍스처 데이터는 필요에 따라 메모리 인터페이스(214)를 통해 판독되고, L2 캐시, 병렬 처리 메모리(204), 또는 시스템 메모리(104)로부터 인출된다. 텍스처 유닛(315)은 내부 캐시에 텍스처 데이터를 저장하도록 구성될 수 있다. 일부 실시예들에서, 텍스처 유닛(315)은 L1 캐시(320)에 연결되고, 텍스처 데이터는 L1 캐시(320)에 저장된다. 각각의 SPM(310)은 추가 처리를 위해 다른 GPC(208)에 처리된 작업을 제공하기 위해 또는 처리된 작업을 크로스바 유닛(210)을 통해 L2 캐시, 병렬 처리 메모리(204), 또는 시스템 메모리(104)에 저장하기 위해 처리된 작업들을 작업 분배 크로스바(330)에 출력한다. preROP(pre-raster operations)(325)는 SPM(310)으로부터 데이터를 수신하고, 파티션 유닛들(215) 내의 ROP 유닛들에 데이터를 보내고, 컬러 블렌딩을 위한 최적화를 수행하고, 픽셀 컬러 데이터를 조직하고, 어드레스 변환을 수행하도록 구성된다.
- [0037] 본 명세서에 설명된 코어 아키텍처는 예시이며, 변형 및 수정이 가능하다는 것을 알 것이다. 임의의 수의 처리 엔진들, 예를 들어, 프리미티브 엔진들(304), SPM들(310), 텍스처 유닛들(315), 또는 preROP들(325)이 GPC(208) 내에 포함될 수 있다. 또한, 하나의 GPC(208)만이 도시되어 있지만, 실행 거동이 GPC(208)가 특정 처

리 작업을 수신하는 것에 의존하지 않도록 PPU(202)는 유리하게 서로 기능적으로 유사한 임의의 수의 GPC(208)를 포함할 수 있다. 또한, 각각의 GPC(208)는 유리하게는 별개의 독특한 처리 엔진들, L1 캐시들(320) 등을 이용하는 다른 GPC들(208)에 독립적으로 동작한다.

[0038] 도 3b는 본 발명의 일 실시예에 따른, 도 2의 PPU들(202) 중 하나의 PPU 내의 파티션 유닛(215)의 블록도이다. 도시된 바와 같이, 파티션 유닛(215)은 L2 캐시(350), 프레임 버퍼(FB)(355) 및 래스터 연산 유닛(raster operations unit; ROP)(360)을 포함한다. L2 캐시(350)는 크로스바 유닛(210) 및 ROP(360)로부터 수신된 로드 및 저장 동작들을 수행하도록 구성되는 판독/기입 캐시이다. 판독 미스들 및 긴급 회신(writeback) 요청들이 처리를 위해 L2 캐시(350)에 의해 FB(355)에 출력된다. 더티 업데이트들(dirty updates)이 또한 기회주의적인 처리를 위해 FB(355)에 전송된다. FB(355)는 병렬 처리 메모리(204)와 직접 인터페이스함으로써, 판독 및 기입 요청들을 출력하고 병렬 처리 메모리(204)로부터 판독된 데이터를 수신한다.

[0039] 그래픽 애플리케이션들에서, ROP(360)는 스텐실, z 테스트, 블렌딩 등과 같은 래스터 동작들을 수행하는 처리 유닛이고, 그래픽 메모리 내의 저장을 위해 처리된 그래픽 데이터로서 픽셀 데이터를 출력한다. 본 발명의 일부 실시예들에서, ROP(360)는 파티션 유닛(215) 대신에 각각의 GPC(208) 내에 포함되며, 픽셀 판독 및 기입 요청들이 픽셀 프래그먼트(fragment) 데이터 대신에 크로스바 유닛(210)을 거쳐서 전송된다.

[0040] 처리된 그래픽 데이터는 디스플레이 디바이스(110) 상에 표시될 수 있거나, CPU(102)에 의해 또는 병렬 처리 서브시스템(112) 내의 처리 엔티티들 중 하나의 엔티티에 의해 추가 처리를 위해 라우팅될 수 있다. 각각의 파티션 유닛(215)은 래스터 동작들의 처리를 분배하기 위해서 ROP(360)를 포함한다. 일부 실시예들에서, ROP(360)는 메모리에 기입되는 z 또는 컬러 데이터를 압축하고 메모리로부터 판독되는 z 또는 컬러 데이터를 압축해제하도록 구성될 수 있다.

[0041] 이 기술분야의 통상의 기술자들은, 도 1, 도 2, 도 3a 및 도 3b에서 설명된 아키텍처는 결코 본 발명의 범위를 한정하는 것이 아니며, 본 명세서에 교시된 기술들은 본 발명의 범위를 벗어나지 않고 하나 이상의 CPU, 하나 이상의 멀티 코어 CPU, 하나 이상의 PPU(202), 하나 이상의 GPC(208), 하나 이상의 그래픽 또는 특수 목적 처리 유닛들 등(이것들로 한정하지 않음)을 포함하는 임의의 적절하게 구성된 처리 유닛에 구현될 수 있다는 것을 이해할 것이다.

[0042] 도 4a는 본 발명의 일 실시예에 따른, 중앙집중형 디바이스 가상화 계층(centralized device virtualization layer; CDVL)을 구현하도록 구성된 컴퓨터 시스템(400)의 개념도이다. 도시된 바와 같이, 컴퓨터 시스템(400)은 복수의 운영 체제(OS)(402A-402N), 하이퍼바이저(404), 중앙 처리 유닛(CPU)(406), 그래픽 처리 장치(GPU)(408), 메모리 브리지(410), 메모리(412), 입출력(I/O) 관리 모듈(414), 및 주변 하드웨어 디바이스들(416A-416N)을 포함한다. I/O 관리 모듈(414)은 주변 하드웨어 디바이스들(416A-416N)에 결합되며, 중앙집중형 디바이스 가상화 계층(CDVL)(426) 및 집적 하드웨어 디바이스들(420A-420N)을 포함한다. CPU(406), GPU(408), 주변 하드웨어 디바이스들(416A-416N) 및 집적 하드웨어 디바이스들(420A-420N)은 각각 메모리(412)에 저장된 하나 이상의 드라이버(424)와 연관된다. 드라이버들(424)은 CPU(406) 또는 GPU(408)에 의해 실행될 때, 프로그램 명령어들을 상이한 유형들의 머신 코드로 번역하는 소프트웨어 프로그램들이다. 예를 들어, CPU(406)는 GPU(408)와 연관된 드라이버(424)를 실행하여, 프로그램 명령어들을 GPU(408)에 사용하도록 고안된 머신 코드로 번역할 수 있다.

[0043] CPU(406)는 컴퓨터 시스템(400)의 주 프로세서이며, 소프트웨어 프로그램들을 실행하여 컴퓨터 시스템(400)의 다양한 기능을 구현하도록 구성된다. 이들 소프트웨어 프로그램들은 OS들(402A-402N), 하이퍼바이저(404) 및 드라이버들(424)을 포함한다. OS들(402A-402N)은 각각의 컴퓨팅 환경과 연관된 애플리케이션들을 실행하는 데 이용될 수 있는 상이한 컴퓨팅 환경들을 사용자에게 제공한다. 예를 들어, OS(402A)는 Microsoft®

Office와 같은 프로그램들로의 액세스를 사용자에게 제공하는 운영 체제인 Windows®

XP일 수 있고, OS(402B)는 FTP(file transfer protocol) 서버로서 기능하는 UNIX 기반 운영 체제일 수 있다. 하이퍼바이저(404)는 하기에서 더 상세하게 설명되는 바와 같이, 충돌을 피하도록 상이한 OS들(402A-402N)의 동작들을 조정하도록 구성된다. 드라이버들(424)은 OS들(402A-402N) 및 OS들(402A-402N) 상에서 실행되는 애플리케이션들이 CPU(406)의 다양한 기능에 액세스가능하게 할 수 있다. OS들(402A-402N), 하이퍼바이저(404) 및 드라이버들(424)은 메모리(412)에 저장될 수 있거나, 또는, 대안적으로, 외부 메모리 유닛(도시되지 않음)에 저장될 수 있다.

- [0044] GPU(408)는 CPU(406)의 처리 능력을 보충하기 위해 그래픽 처리 동작들을 수행하는 코프로세서이다. GPU(408)는 상이한 처리 작업을 수행하기 위해 GPU(408) 내의 다양한 하드웨어 컴포넌트(도시되지 않음)를 구성하는 소프트웨어 프로그램들을 실행할 수 있다. GPU(408)는 구성된 하드웨어 컴포넌트들을 이용하여 그래픽 데이터를 처리할 수 있다. 드라이버들(424)과 같은 GPU(408)에 액세스 가능한 소프트웨어 프로그램들은 메모리(412)에 저장될 수 있거나, 또는, 대안적으로, 외부 메모리 유닛에 저장될 수 있다.
- [0045] CPU(406) 및 GPU(408)는 메모리 브리지(410)에 결합된다. 일 실시예에서, 메모리 브리지(410)는 노스브리지 칩이다. 메모리 브리지(410)는 메모리(412)에 결합된다. CPU(406) 및 GPU(408)는 메모리 브리지(410)를 통해 메모리(412)로부터 데이터를 판독하거나 메모리(412)에 데이터를 기록할 수 있다. 이 데이터는 소프트웨어 프로그램들뿐만 아니라, 그래픽 데이터와 같은 다른 유형의 데이터도 포함할 수 있다.
- [0046] 메모리 브리지(410)는 I/O 관리 모듈(414)에 결합된다. I/O 관리 모듈(414)은 주변 하드웨어 디바이스들(416A-416N) 및 집적 하드웨어 디바이스들(420A-420N)이 CPU(406), GPU(408) 및 이들 처리 장치들에 의해 실행되는 소프트웨어 프로그램들에 액세스할 수 있게 한다. 일 실시예에서, I/O 관리 모듈(414)은 사우스브리지 칩이다. 주변 하드웨어 디바이스들(416A-416N)은 예를 들어, USB(universal serial bus) 커넥터 또는 파이어와이어 커넥터와 같은 커넥터를 이용하여 I/O 관리 모듈(414)에 결합될 수 있다. 주변 하드웨어 디바이스들(416A-416N)은 그중에 키보드, 마우스, 프린터, 스캐너, 디스크 드라이브, 플래시 드라이브, 테이프 드라이브, 마이크 로폰, 스피커, 컴퓨터 모니터, 디지털 비디오 카메라, 또는 다른 컴퓨터 시스템을 포함한다.
- [0047] 집적 하드웨어 디바이스들(420A-420N)은 I/O 관리 모듈(414)을 저장하고 컴퓨터 시스템(400)에 다양한 시스템 기능을 제공하는 동일한 칩 상에 위치한다. 집적 하드웨어 디바이스들(420A-420N)은 그중에 실시간 클록, APM(advanced power management) 모듈, PCI(peripheral component interface) 버스, PCI 익스프레스(PCIe) 버스, DMA(direct memory access) 컨트롤러, 인터럽트 컨트롤러, 베이스보드 관리 컨트롤러, PATA(peripheral advanced technology attachment) 컨트롤러, SATA(serial advanced technology attachment) 컨트롤러, 또는 불 휘발성 BIOS(basic input/output system) 메모리를 포함한다.
- [0048] 집합적으로 "하드웨어 디바이스들"로 불리는, 주변 하드웨어 디바이스들(416A-416N) 및 집적 하드웨어 디바이스들(420A-420N)은 메모리 브리지(410) 및 I/O 관리 모듈(414)을 통해 OS들(402A-402N)에 액세스 가능하다. 컴퓨터 시스템(400)은 OS들(402A-402N)이 하드웨어 디바이스들에 액세스할 수 있도록 소프트웨어(SW) 가상화 및/또는 하드웨어(HW) 가상화를 구현할 수 있다.
- [0049] 도 4b는 본 발명의 일 실시예에 따른, CDVL(426)을 구현하도록 구성된 컴퓨터 시스템(401)의 개념도이다. 도시된 바와 같이, 컴퓨터 시스템(401)은 도 4a에서 설명된 컴퓨터 시스템(400)과 동일한 다수의 컴포넌트를 포함한다. 컴퓨터 시스템(401)은 또한 주변 HW 가상화 디바이스들(417A-417N) 및 집적 HW 가상화 디바이스들(421A-421N)을 포함한다. SW 가상화가 특정 하드웨어 디바이스에 대해 구현될 때, 하이퍼바이저(404)는 하드웨어 디바이스와 연관된 자원들을 여러 부분으로 분할하여 하이퍼바이저(404) 내에 포함되는 SW 가상화 디바이스(422)를 생성한다. SW 가상화 디바이스(422)의 각각의 부분은 가상화되는 하드웨어 디바이스에 의해 제공된 자원들의 단편(fraction)을 나타낸다. 하이퍼바이저(404)는 그 다음에 OS들(402A-402N)의 각각에 SW 가상화 디바이스(422)의 상이한 부분을 할당할 수 있다.
- [0050] 하드웨어 디바이스는 그 하드웨어 디바이스의 복수의 인스턴스(instance)를 애드버타이즈함으로써 HW 가상화를 구현할 수 있다. 주변 하드웨어 디바이스들(416A-416N) 및/또는 집적 하드웨어 디바이스들(420A-420N)은 HW 가상화를 구현할 수 있다. 가상화되는 각각의 주변 하드웨어 디바이스는 주변 HW 가상화 디바이스들(417A-417N)의 세트로서 표현된다. 가상화되는 각각의 집적 하드웨어 디바이스는 집적 HW 가상화 디바이스들(421A-421N)의 세트로서 표현된다.
- [0051] 집합적으로 "HW 가상화 디바이스들"이라고 불리는, 주변 HW 가상화 디바이스들(417A-417N)의 세트들 및 집적 HW 가상화 디바이스들(421A-421N)의 세트들은 각각 가상화되는 단일 하드웨어 디바이스의 복수의 인스턴스를 나타낸다. 하이퍼바이저(404)는 OS들(402A-402N)의 각각에 가상화되는 하드웨어 디바이스의 상이한 인스턴스를 할당할 수 있다. 상이한 OS들이 HW 가상화 디바이스들의 각각에 액세스할 때, 가상화되는 하드웨어 디바이스는 동작이 계속될 수 있게 하기 전에 OS들 사이의 충돌을 해결한다.
- [0052] 주변 하드웨어 디바이스들(416A-416N), 주변 HW 가상 디바이스들(417A-417N), 집적 하드웨어 디바이스들(420A-420N), 집적 HW 가상 디바이스들(421A-421N), 및 SW 가상 디바이스들(422)은 이하 "열거된 디바이스들"로서 총체적으로 지칭되며, 컴퓨터에 대해 광범위한 보충 기능들 및 OS들(402A-402N)을 제공한다. 이들 보충

기능들은, 인쇄, 음향 데이터 녹음, 마우스로부터의 위치 입력 수신 및 그외의 기능들과 같은 각각의 하드웨어와 연관된 기능들을 포함한다. OS들(402A-402N)은 열거된 디바이스들에 액세스 요청을 제출함으로써 이들 기능들을 액세스한다. 액세스 요청들은 열거된 디바이스들을 이용하여 수행될 상이한 액세스 동작들을 지정한다. 액세스 동작들은 구성 동작들 또는 MMIO(memory-mapped I/O) 동작들을 포함한다. 구성 동작은 특정 작업을 위하여 열거된 디바이스를 구성한다. MMIO 동작은 열거된 디바이스의 지정된 어드레스 공간을 이용하여 I/O 동작들을 수행한다.

[0053] OS들(402A-402N)에 의해 전송된 액세스 요청들 및 대응하는 액세스 동작들은 CDVL(426)에 의해 관리된다. CDVL(426)은 OS들(402A-402N) 사이에서 조정하고, 열거된 디바이스들을 이용하여 OS들(402A-402N)이 액세스 동작들을 수행하게 하도록 구성된 기능 유닛이다. CDVL(426)은 OS들(402A-402N)에 의해 전송된 액세스 요청들을 인터셉트한다. CDVL(426)은 열거된 디바이스들에 의해 제공된 자원들을 OS들(402A-402N)에 할당하고, 열거된 디바이스들을 액세스하는 OS들 사이의 충돌들을 회피하도록 기능한다. CDVL(426)은 또한 각각의 열거된 디바이스에 의해 전송된 인터럽트들을 인터셉트하고, 그 인터럽트들을 OS로 라우팅한다.

[0054] HW 가상화가 특정 하드웨어 디바이스를 위해 구현되는 경우, CDVL(426)은 OS들과 HW 가상화 디바이스의 각각의 인스턴스 간의 액세스 동작들을 조정한다. 그리고 나서 가상화되는 하드웨어 디바이스는 물리적인 자원을 할당하여 OS들 사이의 충돌들을 회피한다. SW 가상화가 특정 하드웨어 디바이스를 위해 구현되는 경우, 하이퍼바이저(404)는 OS들(402A-402N)에 의해 이루어진, SW 가상화 디바이스(422)로의 액세스 요청들을 인터셉트한다. 하이퍼바이저(404)는 액세스 요청들을 CDVL(426)에 전송한다. 그리고 나서 CDVL(426)은 SW 가상화되는 하드웨어 디바이스를 이용하여 액세스 요청들을 조정한다.

[0055] 일 실시예에서, CDVL(426)은 I/O 관리 모듈(414) 내에 위치한 하드웨어 유닛으로서 구현된다. 다른 실시예에서, CDVL(426)은 하드웨어 및 마이크로컨트롤러의 조합으로서 구현된다. 또 다른 실시예에서, CDVL(426)은 I/O 관리 모듈(414)과 연관된 메모리에 저장된 소프트웨어 프로그램으로서 구현되며, I/O 관리 모듈(414)에 포함된 마이크로컨트롤러에 의해 실행된다. 메모리(412), 외부 메모리 유닛, 또는 I/O 관리 모듈(414)에 위치될 수 있는 CDVL 셋업 파일(428)은 소프트웨어 구현형 CDVL(426)이 I/O 관리 모듈(414)에 저장되게 한다.

[0056] 또 다른 실시예에서, 메모리 브리지(410) 및 I/O 관리 모듈(414)은 경계(430) 내에 존재하는 단일 칩 상에 집적되고, CDVL(426)은 OS들(402A-402N)이 메모리(412)와 GPU(408)를 이용하여 액세스 동작들을 수행하게 한다. 추가의 실시예에서, 메모리 브리지(410) 및 I/O 관리 모듈(414)은 분리된 칩들이며, OS들(402A-402N)은 CDVL(426)을 통해서만 하드웨어 디바이스들을 액세스할 수 있다.

[0057] CDVL(426)은 OS들(402A-402N) 사이의 충돌들을 하나 이상의 허가 리스트들을 유지함으로써 회피한다. 각각의 허가 리스트들은 어느 열거된 디바이스들로 특정 OS가 액세스하도록 허가되었는지를 나타내는 참조 테이블을 포함한다. CDVL(426)은 OS와 연관된 허가 리스트에 기초하여 이들 열거된 디바이스들로의 액세스를 허가한다. 일 실시예에서, 허가 리스트들은, 오직 하나의 OS만이 특정 시간에 특정 열거된 디바이스를 액세스하는 것이 허가되도록 CDVL(426)에 의해 생성되므로, 잠재적인 충돌들을 회피한다.

[0058] CDVL(426)은 또한 열거된 디바이스를 이용하도록 허가된 OS들의 어드레스 공간으로 각각의 열거된 디바이스의 어드레스 공간을 맵핑하는 VAM들(virtual address maps)을 생성한다. VAM들은 액세스 요청들이 열거된 디바이스들과 연관된 적절한 어드레스로 라우팅되게 하고, 또한 액세스 동작들이 열거된 디바이스의 관련된 어드레스 공간 내에서 수행되게 한다.

[0059] 또한, CDVL(426)은 인터럽트 맵들을 생성한다. CDVL(426)은 인터럽트 맵을 이용하여, 열거된 디바이스로부터, 열거된 디바이스를 이용하도록 허가된 OS들로 전송될 수 있는 인터럽트들을 라우팅한다.

[0060] CDVL(426)은 컴퓨터 시스템(400)의 스타트 업 동안에 활성화된다. CDVL(426)은 OS들(402A-402N)을 식별하고 하드웨어 디바이스들 및 가상화 디바이스들을 열거한다. CDVL(426)은 또한 전술한 바와 같이 허가 리스트들을 생성하고, VAM들 및/또는 인터럽트 맵들을 생성할 수 있다. CDVL(426)이 개시되면, OS들(402A-402N)은 열거된 디바이스들을 액세스할 수 있다.

[0061] VAM들 및 인터럽트 맵들은 디바이스 열거 동안에 생성될 수 있거나, 대안적으로, 허가 리스트가 업데이트되는 경우 동적으로 생성될 수 있다. 예를 들어, 컴퓨터 시스템(400)의 동작 동안에 OS가 열거된 디바이스를 이용하는 허가를 획득하면, CDVL(426)은 OS-열거 디바이스 쌍과 연관된 인터럽트 맵 및 VAM을 동적으로 생성할 수 있다.

- [0062] OS가 열거된 디바이스들 중 하나에 대한 액세스 요청을 전송하는 경우, OS는 OS를 식별하는 액세스 요청 내에 태그를 포함한다. CDVL(426)이 액세스 요청을 인터셉트하는 경우, CDVL(426)은 태그를 조사하고, 열거된 디바이스와 연관된 허가 리스트 상에 OS가 포함되어 있는지의 여부를 결정함으로써, OS가 열거된 디바이스를 액세스하도록 허가되는지의 여부를 결정한다. OS가 열거된 디바이스를 이용하도록 허가되지 않으면, CDVL(426)은 하이퍼바이저(404)에 통지한다. 그리고 나서 하이퍼바이저(404)는 액세스 동작을 거부하기 위한 적절한 액션을 취할 수 있다.
- [0063] OS가 열거된 디바이스를 이용하도록 허가되면, CDVL(426)은 열거된 디바이스의 어드레스 공간을 OS의 어드레스 공간으로 맵핑하는 VAM을 조사한다. CDVL(426)은 액세스 요청을 분석하고, OS에 의해 요청된 어드레스들의 범위를 지정하는 정보를 추출한다. CDVL(426)은 요청된 어드레스들이 맵핑되었는지 또는 맵핑되지 않았는지의 여부를 결정한다. 요청된 어드레스들 중 하나 이상이 맵핑되지 않으면, CDVL(426)은 요청된 어드레스들을 포함하는 새로운 어드레스 맵을 생성한다. 그리고 나서 CDVL(426)은 열거된 디바이스가 액세스 요청에 의해 지정된 액세스 동작을 수행하게 하는 컨텍스트(context)를 열거된 디바이스 내로 로딩한다. 그리고 나서 열거된 디바이스는 액세스 요청을 요구(claim)하고 요청된 액세스 동작을 수행할 수 있다.
- [0064] OS가 열거된 디바이스와 함께 액세스 동작들을 수행하는 경우, 열거된 디바이스는 인터럽트들을 발행할 수 있다. CDVL(426)은 이러한 인터럽트들을 인터셉트하고, 인터럽트 맵들을 이용하여 인터럽트들을 OS의 어드레스 공간 내의 정확한 어드레스로 전송한다. 그 후, OS는 인터럽트들을 처리하기 위해 적절한 단계들을 취할 수 있다. 이러한 단계들은 CDVL(426)을 통해 추가의 액세스 요청들을 열거된 디바이스로 전송하는 것을 포함할 수 있다.
- [0065] 컴퓨터 시스템(401)의 동작 중에, 추가의 하드웨어 디바이스들이 I/O 관리 모듈(414)에 부착될 수 있다. 추가의 하드웨어 디바이스가 I/O 관리 모듈(414)에 부착되는 경우, CDVL(426)은 추가의 하드웨어 디바이스를 열거하고, OS들(402A-402N) 중 어느 것이 추가의 하드웨어 디바이스를 이용하도록 허가되었는지를 결정한다. 그 후, OS들(402A-402N)이 추가의 하드웨어 디바이스에 의해 제공된 자원들을 액세스할 수 있도록, CDVL(426)은 추가의 VAM들 및 추가의 인터럽트 맵들을 생성할 수 있다.
- [0066] CDVL(426)은 또한 컴퓨터 시스템(401)의 동작 중에 추가의 CPU들이 메모리 브리지(410)에 부착되는 것을 허용하도록 구성된다. 추가의 CPU들은 다수의 OS들을 실행할 수 있다. 각각의 CPU는 상이한 하이퍼바이저를 실행하여 그 CPU에 의해 실행되는 OS들을 관리할 수 있다. 이러한 구성은 이하의 도 5에서 설명된다.
- [0067] 도 5는 본 발명의 일 실시예에 따른, CDVL(426)을 구현하도록 구성된 멀티-프로세서 컴퓨터 시스템(500)의 개념도이다. 도시된 바와 같이, 멀티-프로세서 컴퓨터 시스템(500)은 도 4b에 도시된 바와 같이, 컴퓨터 시스템(401)과 동일한 컴포넌트들을 다수 포함한다.
- [0068] 또한, 멀티-프로세서 컴퓨터 시스템(500)은 CPU(502), 하이퍼바이저(504), OS들(506A-506N), CPU(512), 하이퍼바이저(514), 및 OS들(516A-516N)을 포함한다. CPU(502) 및 CPU(512)는 도 4b에 도시된 CPU(406)와 유사한 병렬 처리 유닛들이다. CPU(502) 및 CPU(512)는 소프트웨어 프로그램들을 실행하여 멀티-프로세서 컴퓨터 시스템(500)의 기능을 구현한다. 예를 들어, CPU(502) 및 CPU(512)는, 실행 시에, 프로그램 명령들을 CPU(502) 및 CPU(512)에 각각 네이티브한 머신 코드로 변환할 수 있는 하나 이상의 드라이버들(424)을 각각 실행할 수 있다.
- [0069] CPU(502)는 하이퍼바이저(504) 및 OS들(506A-506N)을 포함하는 소프트웨어 프로그램들을 실행한다. 하이퍼바이저(504)는 하이퍼바이저(404)와 실질적으로 유사하고, OS들(506A-506N)은 실질적으로 OS들(402A-402N)과 유사하다. 하이퍼바이저(504)는 OS들(506A-506N)의 동작들을 관리하도록 구성된다. OS들(506A-506N)은 메모리 브리지(410) 및 I/O 관리 모듈(414)을 통해 하드웨어 디바이스들을 액세스하도록 구성된다. 하나 이상의 하드웨어 디바이스들이 SW 가상화되는 경우, 하이퍼바이저(504)는 OS들(506A-506N)에 의해 액세스가능한 SW 가상화 디바이스들(508)의 하나 이상의 세트를 포함한다.
- [0070] CPU(502)는 하이퍼바이저(514) 및 OS들(516A-516N)을 포함하는 소프트웨어 프로그램들을 실행할 수 있다. 하이퍼바이저(514)는 하이퍼바이저(504)와 실질적으로 유사하고 OS들(516A-516N)은 OS들(506A-506N)과 실질적으로 유사하다. 하이퍼바이저(514)는 OS들(516A-516N)의 동작들을 관리하도록 구성되고, 하나 이상의 하드웨어 디바이스들이 SW 가상화되는 경우에 SW 가상화 디바이스들(518)의 하나 이상의 세트를 포함할 수 있다.
- [0071] OS들(506A-506N) 및 OS들(516A-516N)은, 각각 하이퍼바이저(508)와 하이퍼바이저(518)를 통해 액세스 요청들을 전송함으로써 하드웨어 디바이스들을 액세스할 수 있다. 하이퍼바이저(508) 및 하이퍼바이저(518)는 이러한 액세스 요청들을 CDVL(426)로 라우팅한다. CDVL(426)은 도 4b에서 기술한 방식과 실질적으로 동일한 방식으로 동

작한다. 따라서, CDVL(426)은, OS들(506A-506N) 및/또는 OS들(516A-516N)이 액세스하도록 허가된 하드웨어 디바이스들을 나타내는 허가 리스트들을 보유한다. CDVL(426)은 또한 하드웨어 디바이스들과 그 하드웨어 디바이스들을 이용하도록 허가된 OS들(506A-506N) 및 OS들(516A-516N)에 대한 인터럽트 맵들과 VAM들을 보유한다.

[0072] 도 6은 본 발명의 일 실시예에 따른, 컴퓨터 시스템 상에서 CDVL(426)을 개시하기 위한 방법 단계들의 흐름도이다. 본 기술분야의 숙련자들은, 방법(600)이 도 1-5의 시스템들과 함께 설명되었지만, 이 방법 단계들을 임의의 순서로 수행하도록 구성된 임의의 시스템이 본 발명의 범주 내에 있다는 것을 이해할 것이다.

[0073] 도시된 바와 같이, 방법(600)은 단계(602)에서 시작하고, 여기서 CDVL(426)이 로딩된다. CDVL(426)이 I/O 관리 모듈(414) 내에 포함된 하드웨어 유닛으로서 구현되는 경우, CDVL(426)을 로딩하는 것은 CDVL(426)을 파워-온(power-on)하는 것을 포함한다. CDVL(426)이 소프트웨어 프로그램으로서 구현되는 경우, CDVL(426)을 로딩하는 것은, CDVL 셋업 파일(428)이, CDVL(426)이 I/O 관리 모듈(414)과 연관된 메모리 유닛에 저장되도록 하는 것을 포함한다. 소프트웨어-구현된 CDVL(426)은 컴퓨터 시스템에 포함된 프로세서 또는 I/O 관리 모듈(414)에 포함된 마이크로컨트롤러에 의해 실행된다.

[0074] 단계 604에서, CDVL(426)은 하드웨어 디바이스들의 각각 및 가상화 디바이스들의 각각을 식별하는 디바이스 리스트를 생성함으로써, 하드웨어 디바이스들 및 가상화 디바이스들을 열거한다. CDVL(426)은 또한 각각의 하드웨어 디바이스 및 각각의 가상화 디바이스와 연관된 속성들을 디바이스 리스트에 포함할 수 있다. 또한, CDVL(426)은 하드웨어 디바이스들 및 가상화 디바이스들의 각각을 검사하여, 이들 디바이스들의 각각을 액세스하기 위해 이용될 수 있는 어드레스들의 범위를 결정할 수 있다. 이러한 정보는 디바이스 리스트에 포함될 수 있다.

[0075] 단계 606에서, CDVL(426)은 OS들(506A-506N) 및/또는 OS들(516A-516N) 중 어느 것이 각각의 하드웨어 디바이스 및 각각의 가상화 디바이스를 액세스하도록 허가되는지를 지정하는 허가 리스트들을 생성한다. 일 실시예에서, 상이한 OS들이 동일한 열거된 디바이스와 함께 동시에 액세스 동작들을 수행하는 것이 방지되도록 허가 리스트들이 생성된다.

[0076] 단계 608에서, CDVL(426)은 각각의 하드웨어 디바이스 및 각각의 가상화 디바이스의 어드레스 공간을, 그 열거된 디바이스를 사용하도록 허가된 OS들의 어드레스 공간으로 맵핑하는 VAM들을 생성한다. 단계 610에서, CDVL(426)은 각각의 열거된 디바이스의 인터럽트들을, 그 열거된 디바이스를 사용하도록 허가된 OS들의 어드레스 공간으로 라우팅하는데 사용될 수 있는 인터럽트 맵들을 생성한다.

[0077] 도 7은 본 발명의 일 실시예에 따른, 컴퓨터 시스템 상에서 CDVL(426)을 구현하기 위한 방법 단계들의 흐름도이다. 본 기술분야의 숙련자들은, 방법(700)이 도 1-5의 시스템들과 함께 설명되었지만, 이 방법 단계들을 임의의 순서로 수행하도록 구성된 임의의 시스템이 본 발명의 범주 내에 있다는 것을 이해할 것이다.

[0078] 도시한 바와 같이, 방법(700)은 단계 702에서 시작하는데, 이 단계에서 CDVL(426)은 OS로부터 액세스 요청을 수신한다. OS는, 예컨대, OS들(402A-402N) 중 하나, OS들(506A-506N) 중 하나, 또는 OS들(516A-516N) 중 하나일 수 있다. 액세스 요청은 OS를 식별하는 태그를 포함한다. 액세스 요청은 또한 OS가 액세스하기를 원하는 열거된 디바이스, OS가 수행하기를 원하는 액세스 동작, 및 OS가 액세스하기를 원하는 어드레스들을 포함한다.

[0079] 단계 704에서, CDVL(426)은 OS가 액세스 요청에서 지정한 열거된 디바이스에 액세스하는 것을 허가할지 여부를 결정한다. CDVL(426)은 액세스 요청과 연관된 태그를 검사하여 OS, 즉, 그로부터 액세스 요청이 수신되는 OS를 결정한다. CDVL(426)은 또한 액세스 요청을 검사하여 어떠한 열거된 디바이스가 요청되는지를 결정한다. 다음으로, CDVL(426)은 요청된 열거된 디바이스와 연관된 허가 리스트에 액세스한다. 허가 리스트는 전술한 바와 같이 그 열거된 디바이스를 이용하도록 허가되는 OS들을 지정한다. CDVL(426)은 OS가 허가 리스트에 포함되어 있는지의 여부 및, 그에 따라, OS가 요청된 열거된 디바이스에 액세스하는 것이 허가되는지의 여부를 결정한다.

[0080] OS가 허가 리스트에 포함되어 있지 않으면, OS는 그 열거된 디바이스에 액세스하는 것이 허가되지 않고, 방법은 단계 706으로 진행한다. 단계 706에서, CDVL(426)은 OS가 열거된 디바이스에 액세스하는 것을 거부하고, 방법(700)은 종료한다.

[0081] OS가 허가 리스트에 포함되어 있으면, 방법(700)은 단계 708로 진행한다. 단계 708에서, CDVL(426)은 액세스 요청에 맵핑되지 않은 어드레스가 포함되어 있는지 여부를 결정한다. CDVL(426)은 액세스 요청을 검사하여 OS가 액세스하려고 하는 어드레스를 결정한다. CDVL(426)은 또한 OS와 연관된 VAM 및 열거된 디바이스에 액세스한다. CDVL(426)은 액세스 요청에 포함된 어드레스를 VAM과 비교하여, 지정된 어드레스 중 어느 것이 VAM에 포

함되어 있지 않은지를 결정하는데, 이는 이들 어드레스가 맵핑되지 않은 것임을 나타낸다.

- [0082] 액세스 요청이 맵핑되지 않은 어드레스를 포함하고 있지 않으면(즉, 액세스 요청에서 지정된 모든 어드레스가 VAM에 포함되어 있으면), 방법(700)은 단계 712로 진행한다.
- [0083] 액세스 요청에서 지정된 임의 어드레스가 맵핑되지 않은 것이면, 방법(700)은 단계 710으로 진행한다. 단계 710에서, CDVL(426)은 맵핑되지 않은 어드레스를 열거된 디바이스의 어드레스 공간으로 맵핑함으로써 VAM을 업데이트한다. 다음으로, 방법(700)은 단계 712로 진행한다.
- [0084] 단계 712에서, CDVL(426)은 요청된 열거된 디바이스와의 요청된 액세스 동작의 계속을 허용한다. CDVL(426)은 VAM을 이용하여 열거된 디바이스의 어드레스 공간 내의 적절한 어드레스로, OS에 의해 전송될 수 있는 구성 동작 및 MMIO 동작을 라우팅함으로써 액세스 동작을 조정한다. CDVL은 또한 인터럽트 맵을 이용하여 OS의 어드레스 공간 내의 적절한 어드레스로, 열거된 디바이스에 의해 생성된 인터럽트를 라우팅한다. 액세스 동작이 완료하면, 방법(700)은 종료한다.
- [0085] 요약하면, CDVL은 멀티 CPU 컴퓨터 시스템에서 상이한 CPU 상에서 실행되는 OS에 대한 디바이스 가상화를 지원한다. CDVL은 상이한 CPU들을 하나 이상의 열거된 디바이스들과 접속시키는 SB(Southbridge) 칩과 같은 I/O 관리 모듈 내로부터 동작한다. 열거된 디바이스는 하드웨어 디바이스와 가상화 디바이스를 포함한다. CDVL은 열거된 디바이스에 액세스할 수 있는 OS를 포함하는 각각의 열거된 디바이스에 대한 허가 리스트를 생성함으로써, 어느 OS들이 각각의 열거된 디바이스를 이용하도록 허가되는지를 추적한다. 어느 한 OS가 열거된 디바이스를 이용하도록 허가되면, 그 OS는 구성 동작 및/또는 메모리 맵핑된 I/O 동작을 포함할 수 있는, 그 열거된 디바이스와의 액세스 동작을 수행할 수 있다.
- [0086] OS가 특정 열거된 디바이스가 제공하는 자원에 액세스하려고 하는 경우, OS는 OS를 식별하는 태그를 포함하는 액세스 요청을 생성한다. 다음으로, OS는 OS를 관리하는 하이퍼바이저를 통해 CDVL에 액세스 요청을 전송한다. CDVL은 액세스 요청과 연관된 태그를 검사하여, OS가 열거된 디바이스에 액세스하는 것이 허가되는지의 여부를 결정한다. OS가 이용하도록 허가되지 않는 열거된 디바이스에 액세스하려고 하는 경우, CDVL은 OS와 연관된 하이퍼바이저에 통지한다. 다음으로, 하이퍼바이저는 OS가 열거된 디바이스에 액세스하는 것을 거부하는 적절한 단계들을 취할 수 있다.
- [0087] CDVL은 VAM을 이용하여 열거된 디바이스로 구성 동작 및/또는 MMIO 동작을 라우팅한다. VAM은 각각의 열거된 디바이스의 어드레스 공간을 열거된 디바이스에 대한 액세스가 허가되는 OS의 어드레스 공간으로 맵핑하는 것을 제공한다. CDVL은 디바이스 열거 동안 또는 런타임 동안에 VAM을 생성할 수 있다.
- [0088] OS가 CDVL에 액세스 요청을 CDVL에 전송하고, CDVL이 액세스 요청을 허가하면, CDVL은 요청된 어드레스가 맵핑된 것을 보장하기 위해서 관련 VAM을 검사한다. 요청된 어드레스가 맵핑되지 않은 것이면, CDVL은 새로운 VAM을 생성할 수 있다. CDVL은 VAM을 이용하여 열거된 디바이스에 대한 정확한 어드레스로 액세스 요청을 라우팅한다. CDVL은 또한 콘텍스트 스위칭이나 디바이스 정지(quiescing)와 같이, 액세스의 계속을 허용하는 다른 액션을 수행할 수 있다.
- [0089] CDVL은 특정 열거된 디바이스로부터의 인터럽트를 열거된 디바이스에 액세스할 수 있는 OS로 맵핑함으로써 열거된 디바이스 인터럽트를 처리한다. 열거된 디바이스가 인터럽트를 발행하면, CDVL은 그 인터럽트를 OS를 관리하는 하이퍼바이저를 통해 OS 어드레스 공간 내의 관련 어드레스로 전송할 수 있다. CDVL은 디바이스 열거 동안 또는 열거된 디바이스 허가 변경 시에 인터럽트 맵을 생성할 수 있다.
- [0090] 유리하게는, CDVL은 다수의 CPU를 포함하는 컴퓨팅 시스템으로 디바이스 가상화를 확장한다. CDVL은 각각의 CPU 상에서 실행되는 OS가 컴퓨터 시스템에 연결되는 열거된 디바이스들의 공동 풀(pool)에 액세스할 수 있도록 한다. CDVL은 전용 가상화 지원을 그러한 지원을 필요로 하는 OS에 제공하기 위해서 SW 가상화 및 HW 가상화와 함께 이용될 수 있다. CDVL의 다른 이점은 추가의 CPU를 메모리 브리지에 연결함으로써 컴퓨터 시스템에 추가의 CPU를 추가할 수 있다는 점이다. 추가의 CPU 각각이 하나 이상의 OS를 실행하는 경우, 메모리 브리지에 연결된 CPU 상에서 실행되는 OS의 각각은 CDVL을 통해 열거된 디바이스에 액세스할 수 있다.
- [0091] 본 발명의 일 실시예는 컴퓨터 시스템과 함께 사용하기 위한 프로그램 제품으로서 구현될 수 있다. 프로그램 제품의 프로그램(들)은 실시예들의 기능들(본원에 기재된 방법들 포함)을 정의하고 각종 컴퓨터 판독가능 저장 매체 상에 포함될 수 있다. 예시적인 컴퓨터 판독가능 저장 매체는: (i) 정보가 영구적으로 저장되는 비기록가능 저장 매체(예를 들어, CD-ROM 드라이브에 의해 판독가능한 CD-ROM 디스크들, 플래시 메모리, ROM 칩들 또는 임의 유형의 고체 상태 불휘발성 반도체 메모리 등의 컴퓨터 내의 판독 전용 메모리 디바이스들); 및 (ii)

변경가능한 정보가 저장되는 기록가능 저장 매체(예를 들어, 하드 디스크 드라이브 또는 디스켓 드라이브 내의 플로피 디스크들 또는 임의 유형의 고체 상태 랜덤 액세스 반도체 메모리)를 포함하나 이에 제한되지 않는다.

[0092] 본 발명은 특정 실시예들을 참조하여 진술하였다. 그러나, 이 기술분야의 통상의 기술자들은 첨부된 특허청구 범위에 개시된 바와 같이 본 발명의 더 넓은 사상 및 범위로부터 벗어나지 않고 본 발명에 대해 다양한 수정 및 변경이 이루어질 수 있다는 것을 이해할 것이다. 그러므로, 진술한 설명 및 도면들은 한정적 의미가 아닌 예시로서 여겨져야 한다.

도면의 간단한 설명

[0093] 도 1은 본 발명의 하나 이상의 양상들을 구현하도록 구성된 컴퓨터 시스템을 도시하는 블록도.

[0094] 도 2는 본 발명의 일 실시예에 따른, 도 1의 컴퓨터 시스템을 위한 병렬 처리 서브시스템의 블록도.

[0095] 도 3a는 본 발명의 일 실시예에 따른, 도 2의 PPU(parallel processing unit)들 중 하나 내의 GPC(general processing cluster)의 블록도.

[0096] 도 3b는 본 발명의 일 실시예에 따른, 도 2의 PPU들 중 하나 내의 파티션 유닛의 블록도.

[0097] 도 4a 및 도 4b는 본 발명의 일 실시예에 따른, CVDL(centralized device virtualization layer)를 구현하도록 구성된 컴퓨터 시스템의 블록도.

[0098] 도 5는 본 발명의 일 실시예에 따른, CVDL을 구현하도록 구성된 멀티 프로세서 컴퓨터 시스템의 개념도

[0099] 도 6은 본 발명의 일 실시예에 따른, 컴퓨터 시스템에서 CVDL을 개시하기 위한 방법 단계들의 흐름도.

[0100] 도 7은 본 발명의 일 실시예에 따른, 컴퓨터 시스템에서 CVDL을 구현하기 위한 방법 단계들의 흐름도.

[0101] <도면의 주요 부분에 대한 부호의 설명>

[0102] 408 : GPU

[0103] 410 : 메모리 브리지

[0104] 412 : 메모리

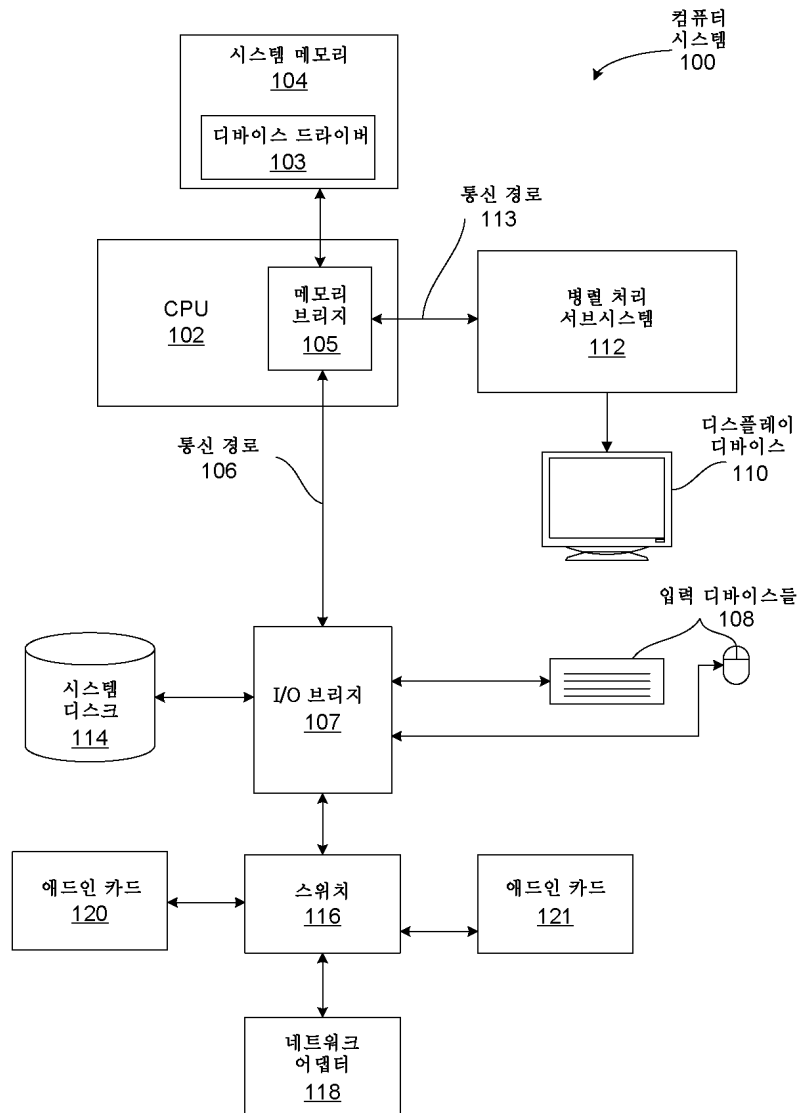
[0105] 502, 512 : CPU

[0106] 504, 514 : 파이프라이저

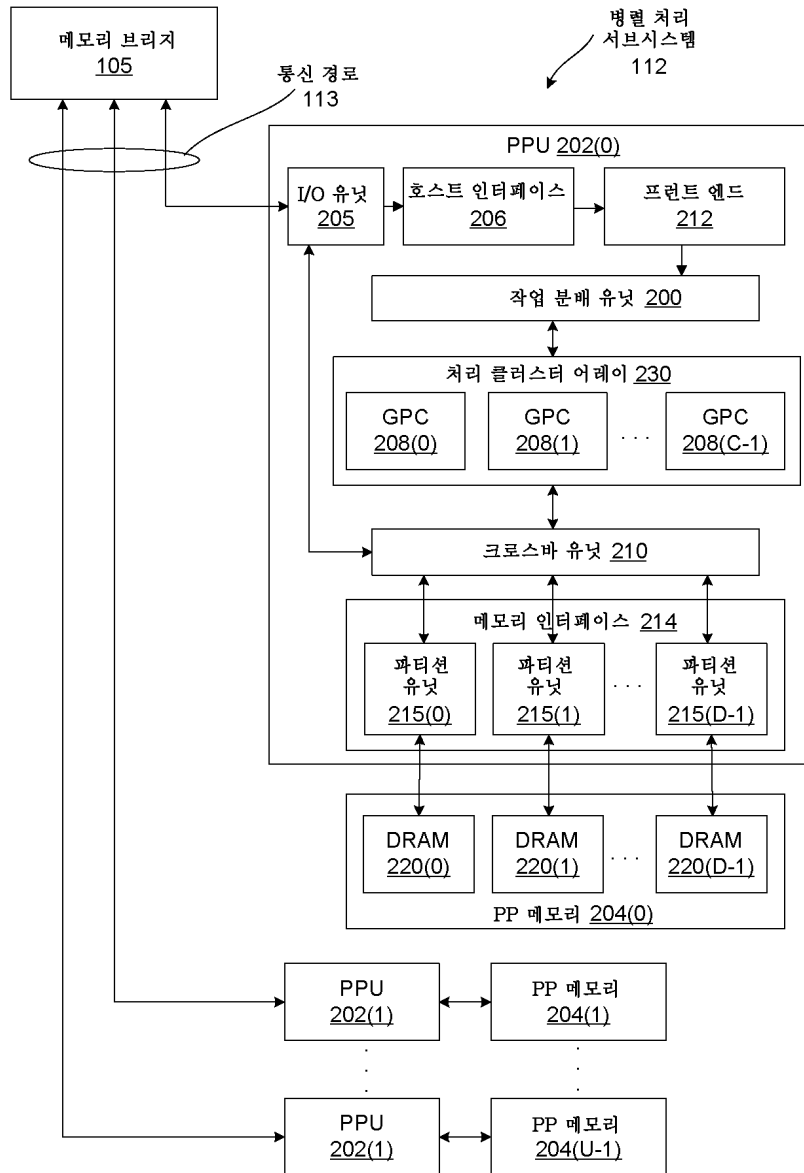
[0107] 506A-506N, 516A-516N : OS

도면

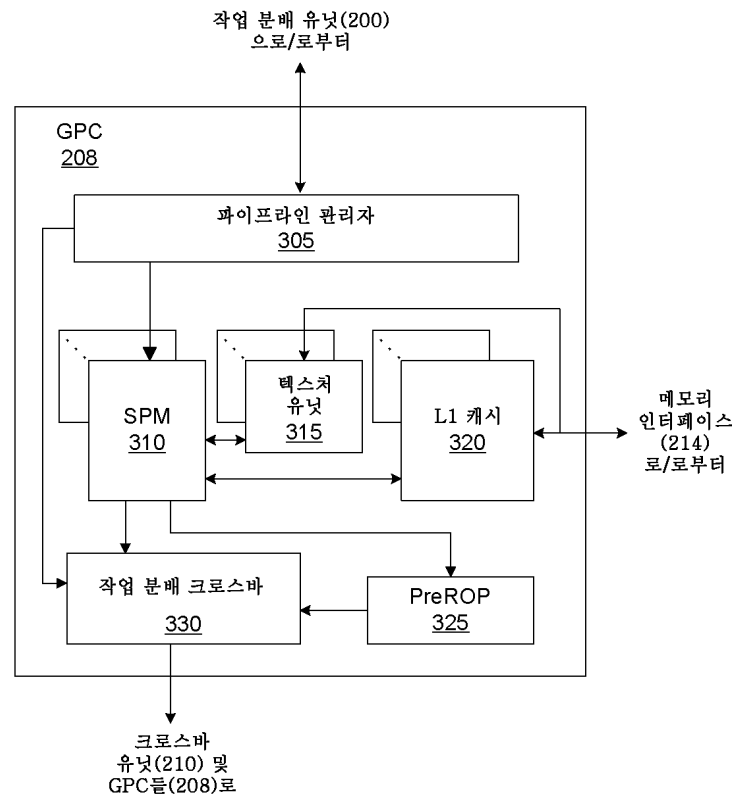
도면1



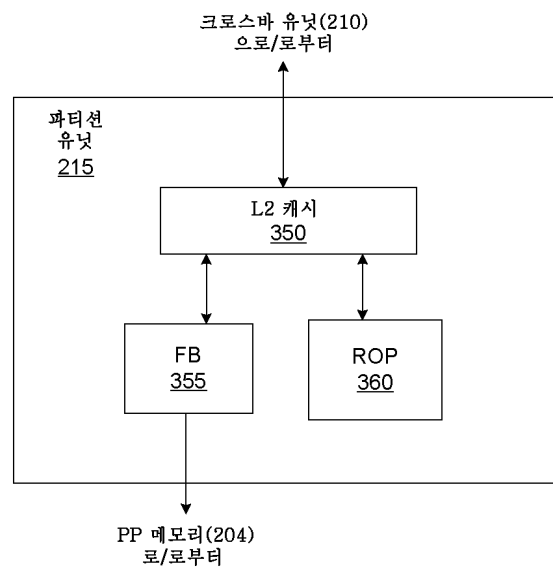
도면2



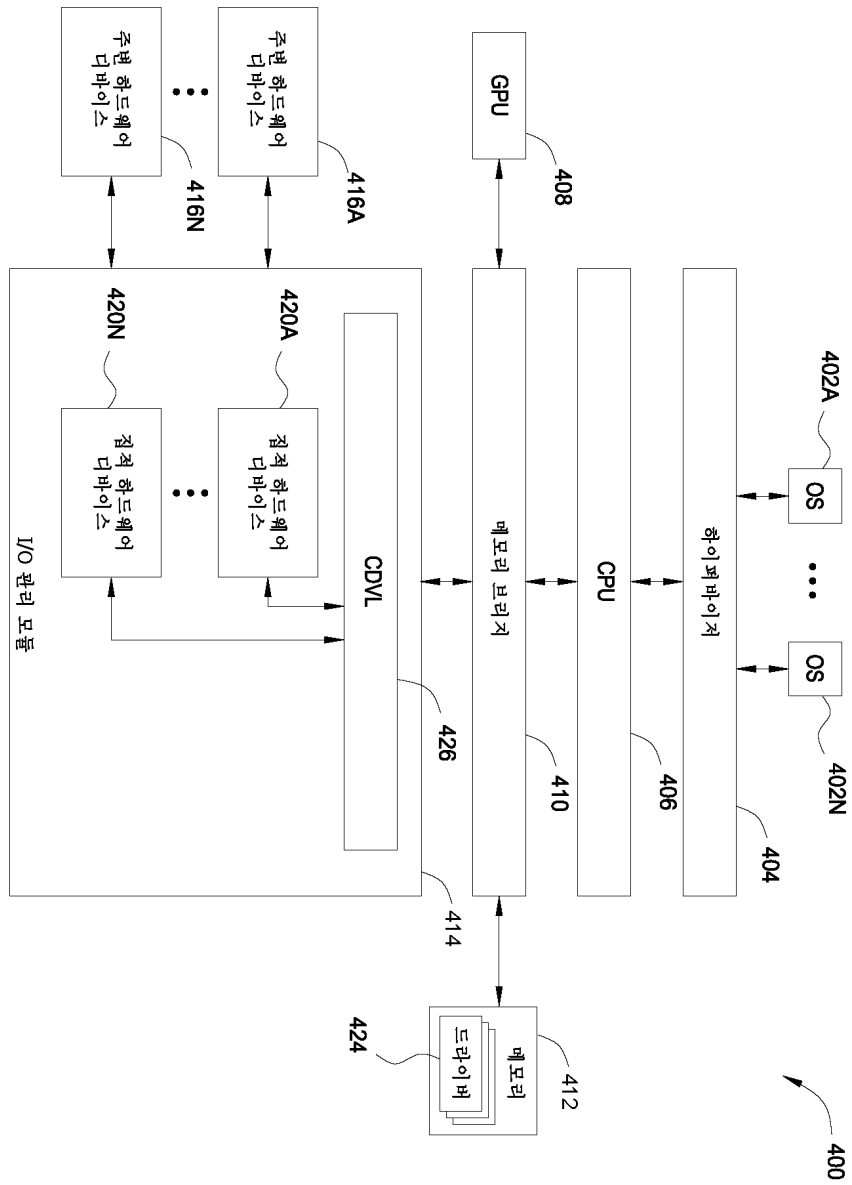
도면3a



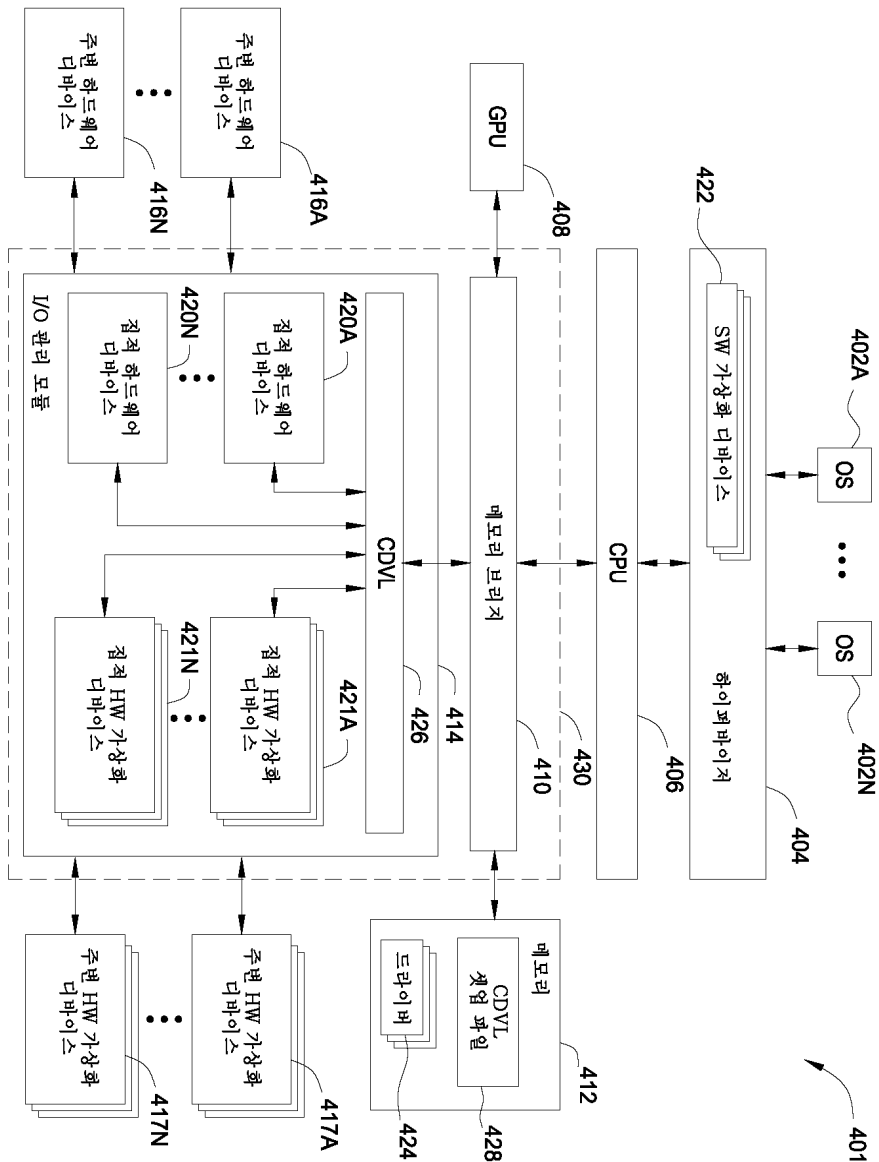
도면3b



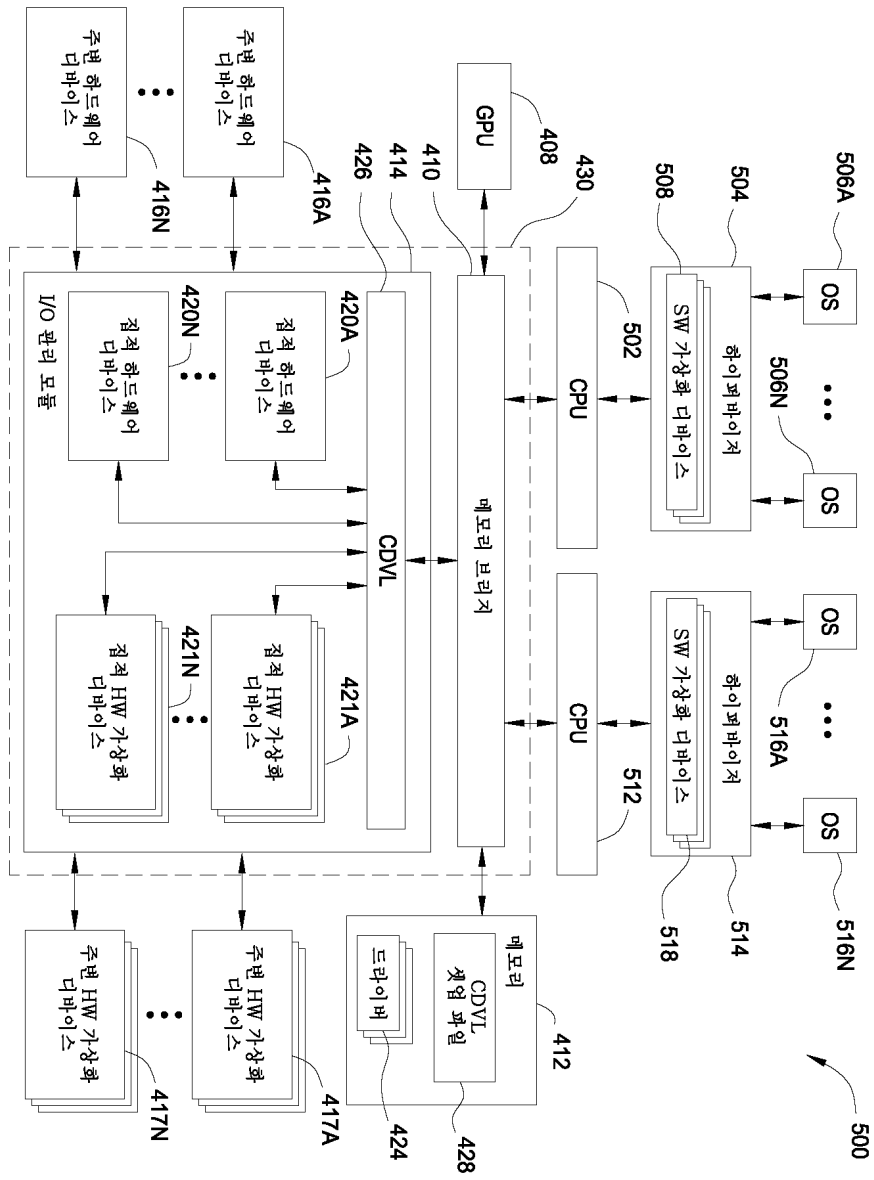
도면4a



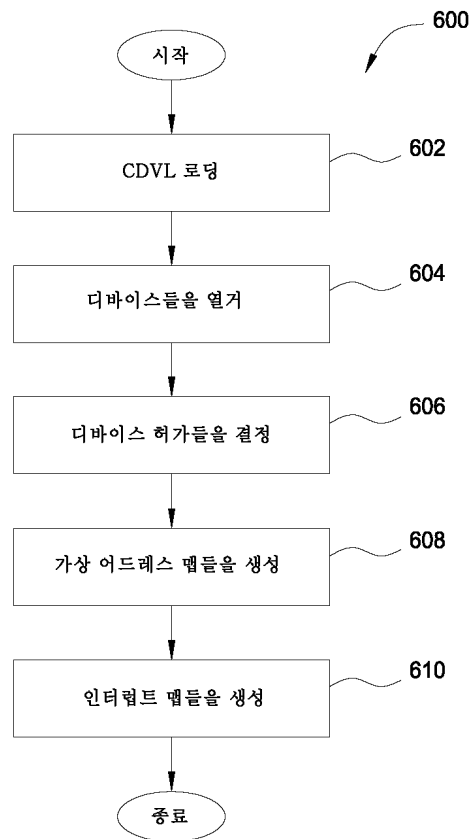
도면4b



도면5



도면6



도면7

