(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2022년08월11일
(11) 등록번호 10-2430982
(24) 등록일자 2022년08월04일

(51) 국제특허분류(Int. Cl.)
G06F 13/16 (2006.01) *G06F 12/02* (2018.01)
G06F 12/06 (2006.01) *G06F 3/06* (2006.01)
(52) CPC특허분류
G06F 13/1694 (2013.01)
G06F 12/0292 (2013.01)
(21) 출원번호 10-2021-0117942
(22) 출원일자 2021년09월03일
심사청구일자 2021년09월03일
(30) 우선권주장
1020210075745 2021년06월10일 대한민국(KR)
(56) 선행기술조사문헌
KR1020190017639 A
KR1020190072404 A
KR1020190100632 A
KR1020200039930 A

(73) 특허권자
삼성전자주식회사
경기도 수원시 영통구 삼성로 129 (매탄동)
연세대학교 산학협력단
서울특별시 서대문구 연세로 50 (신촌동, 연세대학교)
(72) 발명자
정원섭
경기도 수원시 영통구 센트럴타운로 76, 6103동 1101호(이의동, e편한세상 광교)
갈홍주
서울특별시 강남구 영동대로65길 38(대치동)
(뒷면에 계속)
(74) 대리인
리앤목특허법인

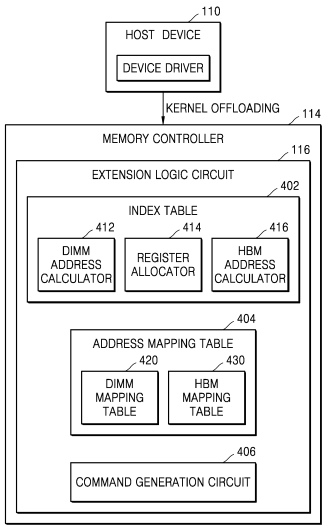
전체 청구항 수 : 총 10 항 심사관 : 김세영

(54) 발명의 명칭 **프로세싱부를 포함하는 이종 메모리 시스템을 억세스하는 데이터 처리 시스템 및 방법**

(57) 요 약

프로세싱부를 포함하는 이종 메모리 시스템을 억세스하는 데이터 처리 시스템 및 방법이 기술된다. 이종 메모리 시스템은 ,메모리 모듈 및 메모리-내-처리(Processing-In-Memory: PIM) 회로를 포함하는 고대역 메모리(HBM)으로 구성되고, 메모리 콘트롤러와 결합된다. 메모리 콘트롤러는 HBM의 PIM 회로에서 연산 작업이 실행될 때, 보더 인덱스 값을 이용하여 연산 작업에 요구되는 데이터 어레이를 메모리 모듈 또는 HBM으로부터 검색하고, 메모리 모듈 및 HBM 각각에 지정된 물리 주소 공간을 사용하는 메모리 모듈 명령어 세트 및 HBM 명령어 세트를 생성한다.

대 표 도 - 도4



(52) CPC특허분류
G06F 12/0638 (2013.01)
G06F 3/0604 (2013.01)
G06F 3/0659 (2013.01)
G06F 3/0661 (2013.01)
(72) 발명자
노원우
서울특별시 강남구 삼성로51길 35, 201동 1202호(대치동, 래미안 대치 팰리스)
이석민
경상북도 고령군 다산면 상곡3길 72
고건
경기도 군포시 수리산로 244, 994동 401호(산본동, 한양백두아파트)

명 세 서

청구범위

청구항 1

시스템에 있어서,

데이터를 저장하도록 구성되는 이종 메모리 시스템, 상기 이종 메모리 시스템은 메모리 모듈 및 메모리-내-처리(Processing-In-Memory: PIM) 회로를 포함하는 고대역 메모리(HBM)를 포함하고;

상기 시스템의 연산 작업들을 실행하도록 구성되는 호스트 프로세서, 상기 호스트 프로세서는 상기 연산 작업들 중 일부가 상기 PIM 회로에서 실행되도록 할당하고; 및

상기 호스트 프로세서에 의해 오프로드된 커널에 따른 연산 작업이 상기 HBM의 상기 PIM 회로에서 실행될 때, 상기 연산 작업에 요구되는 데이터 어레이가 존재하는 메모리를 판단하는 기준이 되는 보더 인덱스 값을 이용하여 상기 데이터 어레이를 상기 메모리 모듈 또는 상기 HBM으로부터 검색하는 메모리 콘트롤러를 포함하는 시스템.

청구항 2

제1항에 있어서,

상기 보더 인덱스 값은 상기 데이터 어레이의 활용 빈도 비율 또는 상기 메모리 모듈과 상기 HBM의 대역폭 비율에 기초하여 설정되는 시스템.

청구항 3

제1항에 있어서,

상기 메모리 콘트롤러는, 상기 호스트 프로세서로부터 상기 오프로드된 커널에 대한 정보를 수신하고,

상기 오프로드된 커널에 대한 정보는 상기 데이터 어레이의 식별 번호와 상기 식별 번호에 해당하는 상기 데이터 어레이에 대한 베이스 어드레스, 인덱스 값 및 데이터 사이즈를 포함하고, 상기 인덱스 값은 상기 베이스 어드레스와의 오프셋을 나타내는 시스템.

청구항 4

제3항에 있어서,

상기 메모리 콘트롤러는 상기 인덱스 값과 상기 보더 인덱스 값을 비교하여 상기 인덱스 값이 상기 보더 인덱스 값 보다 작으면 상기 데이터 어레이가 상기 HBM에 존재하는 것으로 판단하는 시스템.

청구항 5

제4항에 있어서,

상기 메모리 콘트롤러는 HBM 맵핑 테이블을 포함하고, 상기 데이터 어레이에 대한 상기 베이스 어드레스에 대응하는 상기 HBM의 베이스 어드레스를 상기 HBM 맵핑 테이블에서 읽는 시스템.

청구항 6

제5항에 있어서,

상기 메모리 콘트롤러는 상기 HBM의 상기 베이스 어드레스로부터 상기 인덱스 값 만큼 떨어진 위치에 대응하는 제1 어드레스를 계산하고, 상기 제1 어드레스로부터 상기 데이터 사이즈를 곱한 값에 대응하는 제2 어드레스를 계산하고, 상기 HBM의 상기 제1 어드레스로부터 상기 제2 어드레스까지의 물리 주소 공간에 상기 데이터 어레이가 저장되어 있는 것으로 판단하는 시스템.

청구항 7

제6항에 있어서,

상기 메모리 콘트롤러는 상기 HBM에 저장된 상기 데이터 어레이를 이용하여 상기 HBM의 상기 PIM 회로에서 실행되는 연산 처리에 의해 생성되는 중간 결과 값을 저장하는 레지스터를 할당하는 시스템.

청구항 8

제7항에 있어서,

상기 메모리 콘트롤러는 상기 HBM의 상기 PIM 회로에서 상기 연산 처리를 실행하도록 지시하는 HBM 명령어 세트를 생성하고,

상기 HBM 명령어 세트는 오피코드 파라미터, 벡터 사이즈 파라미터, 소스 파라미터, 및 목적지 파라미터를 포함하고,

상기 오피코드 파라미터는 상기 오프로드된 커널의 명령어과 동일하게 표기되고, 상기 벡터 사이즈 파라미터는 상기 호스트 프로세서와 결합되는 DRAM 캐시 라인의 데이터 입출력 수와 상기 데이터 어레이의 데이터 입출력 수와의 상관 관계를 표시하고, 상기 소스 파라미터는 상기 데이터 어레이에 대응하는 상기 HBM의 어드레스로 표기되고, 상기 목적지 파라미터는 상기 할당된 레지스터의 식별 번호를 표기하는 시스템.

청구항 9

제3항에 있어서,

상기 메모리 콘트롤러는 상기 인덱스 값과 상기 보더 인덱스 값을 비교하여 상기 인덱스 값이 상기 보더 인덱스 값 보다 크면 상기 데이터 어레이가 상기 메모리 모듈에 존재하는 것으로 판단하는 시스템.

청구항 10

메모리 모듈 및 메모리-내-처리(Processing-In-Memory: PIM) 회로를 포함하는 고대역 메모리(HBM)를 포함하는 이종 메모리 시스템을 포함하는 시스템에서 구현되는 방법으로서,

호스트 프로세서에 의해 커널을 메모리 콘트롤러에 오프로드하는 단계;

상기 오프로드된 커널의 작업 코드의 실행에 따라 처리될 데이터 어레이가 존재하는 메모리를 판단하는 기준이 되는 보더 인덱스 값을, 상기 메모리 콘트롤러에 의해, 저장하는 단계;

상기 오프로드된 커널에 대한 정보를, 상기 메모리 콘트롤러에 의해, 수신하는 단계, 상기 오프로드된 커널에 대한 정보는 상기 데이터 어레이의 식별 번호와 상기 식별 번호에 해당하는 상기 데이터 어레이에 대한 베이스 어드레스, 인덱스 값 및 데이터 사이즈를 포함하고, 상기 인덱스 값은 상기 베이스 어드레스와의 오프셋을 나타내고;

상기 데이터 어레이의 상기 인덱스 값과 상기 보더 인덱스 값을, 상기 메모리 콘트롤러에 의해, 비교하는 단계;

비교 결과, 상기 인덱스 값이 상기 보더 인덱스 값 보다 작으면 상기 데이터 어레이가 상기 HBM에 존재하는 것으로 판단하고, 상기 인덱스 값이 상기 보더 인덱스 값 보다 크면 상기 데이터 어레이가 상기 메모리 모듈에 존재하는 것으로, 상기 메모리 콘트롤러에 의해, 판단하는 단계;

상기 메모리 모듈에 저장된 상기 데이터 어레이를 이용하여 상기 HBM의 상기 PIM 회로에서 연산 작업을 실행하도록 지시하는 메모리 모듈 명령어 세트를, 상기 메모리 콘트롤러에 의해, 생성하는 단계; 및

상기 HBM에 저장된 상기 데이터 어레이를 이용하여 상기 HBM의 상기 PIM 회로에서 상기 연산 작업을 실행하도록 지시하는 HBM 명령어 세트를, 상기 메모리 콘트롤러에 의해, 생성하는 단계를 포함하고,

상기 보더 인덱스 값은 상기 데이터 어레이의 활용 빈도 비율 또는 상기 메모리 모듈과 상기 HBM의 대역폭 비율에 기초하여 설정되는 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 장치들(apparatuses) 및 방법들(methods)에 관한 것으로서, 더욱 상세하게는 프로세싱부를 포함하는

이종 메모리 시스템을 억세스하는 데이터 처리 시스템 및 방법에 관한 것이다.

배 경 기 술

[0002] 전형적으로 메모리 프로세싱은 단일(single) 메모리에 대해서 수행되도록 구성된다. 메모리 프로세싱의 성능 및 특성이 단일 메모리에 의해 결정될 수 있다. 큰 메모리 용량을 요구하는 어플리케이션 프로세싱은 고용량의 메모리 모듈, 예컨대 DIMM(Dual In-line Memory Module)을 활용할 수 있다. 이에 대해, 높은 대역폭을 요구하는 어플리케이션 프로세싱은 고대역폭의 메모리 장치, 예컨대 HBM(High Bandwidth Memory)를 활용할 수 있다. 이처럼 단일 메모리는 고용량의 DIMM 또는 고대역폭의 HBM으로 활용될 수 있다.

[0003] 고용량의 DIMM으로 구현된 메모리를 활용하는 어플리케이션 프로세싱에서 간헐적으로 고대역폭 메모리 동작이 수행될 수 있다. 이를 지원하기 위해 다수의 DIMM들을 사용하게 되면 어플리케이션의 필요 용량 이상의 메모리 용량이 잉여될(excess) 수 있다. 이와 반대로, 고대역폭의 HBM으로 구현된 메모리를 활용하는 어플리케이션 프로세싱은 가끔의 고용량 메모리 동작을 수행할 수 있는데, 이를 지원하기 위해 다수의 HBM들을 사용하게 되면 어플리케이션이 요구하는 대역폭을 초과하게 되어 어플리케이션 실행 속도가 느려질(slow) 수 있다.

[0004] 이에 따라, 메모리 용량 및 대역폭 둘 다를 지원하기 위해 DIMM 및 HBM으로 구현된 이종 메모리 시스템이 요구된다.

발명의 내용

해결하려는 과제

[0005] 본 발명의 목적은 프로세싱부를 포함하는 이종 메모리 시스템을 억세스하는 데이터 처리 시스템 및 방법을 제공하는 데 있다.

과제의 해결 수단

[0006] 본 발명의 실시예들에 따른 시스템은, 데이터를 저장하도록 구성되는 이종 메모리 시스템, 상기 이종 메모리 시스템은 메모리 모듈 및 메모리-내-처리(Processing-In-Memory: PIM) 회로를 포함하는 고대역 메모리(HBM)를 포함하고; 상기 시스템의 연산 작업들을 실행하도록 구성되는 호스트 프로세서, 상기 호스트 프로세서는 상기 연산 작업들 중 일부가 상기 PIM 회로에서 실행되도록 할당하고; 및 상기 호스트 프로세서에 의해 오프로드된 커널에 따른 연산 작업이 상기 HBM의 상기 PIM 회로에서 실행될 때, 상기 연산 작업에 요구되는 데이터 어레이가 존재하는 메모리를 판단하는 기준이 되는 보더 인덱스 값을 이용하여 상기 데이터 어레이를 상기 메모리 모듈 또는 상기 HBM으로부터 검색하는 메모리 콘트롤러를 포함한다.

[0007] 본 발명의 실시예들에 따른 장치는, 제1 데이터를 저장하도록 구성되는 메모리 모듈; 제2 데이터를 저장하도록 구성되는 고대역 메모리(HBM), 상기 HBM에는 다수의 메모리 다이들이 스택되고, 상기 메모리 다이들 각각은 연산 작업을 수행하는 메모리-내-처리(Processing-In-Memory: PIM) 회로를 포함하고; 및 상기 PIM 회로의 상기 연산 작업에 요구되는 데이터 어레이가 존재하는 메모리를 판단하는 기준이 되는 보더 인덱스 값을 이용하여 상기 데이터 어레이를 상기 메모리 모듈 또는 상기 HBM으로부터 검색하도록 구성되는 메모리 콘트롤러를 포함하고, 상기 메모리 콘트롤러는 상기 데이터 어레이와 관련하여 상기 메모리 모듈에 지정된 물리 주소 공간을 사용하는 메모리 모듈 명령어 세트 및 상기 HBM에 지정된 물리 주소 공간을 사용하는 HBM 명령어 세트를 생성한다.

[0008] 본 발명의 실시예들에 따른 메모리 모듈 및 메모리-내-처리(Processing-In-Memory: PIM) 회로를 포함하는 고대역 메모리(HBM)를 포함하는 이종 메모리 시스템을 포함하는 시스템에서 구현되는 방법은, 호스트 프로세서에 의해 커널을 메모리 콘트롤러에 오프로드하는 단계; 상기 오프로드된 커널의 작업 코드의 실행에 따라 처리될 데이터 어레이가 존재하는 메모리를 판단하는 기준이 되는 보더 인덱스 값을, 상기 메모리 콘트롤러에 의해, 저장하는 단계; 상기 오프로드된 커널에 대한 정보를, 상기 메모리 콘트롤러에 의해, 수신하는 단계, 상기 오프로드된 커널에 대한 정보는 상기 데이터 어레이의 식별 번호와 상기 식별 번호에 해당하는 상기 데이터 어레이에 대한 베이스 어드레스, 인덱스 값 및 데이터 사이즈를 포함하고, 상기 인덱스 값은 상기 베이스 어드레스와의 오프셋을 나타내고; 상기 데이터 어레이의 상기 인덱스 값과 상기 보더 인덱스 값을, 상기 메모리 콘트롤러에 의해, 비교하는 단계; 비교 결과, 상기 인덱스 값이 상기 보더 인덱스 값 보다 작으면 상기 데이터 어레이가 상기 HBM에 존재하는 것으로 판단하고, 상기 인덱스 값이 상기 보더 인덱스 값 보다 크면 상기 데이터 어레이가 상기 메모리 모듈에 존재하는 것으로, 상기 메모리 콘트롤러에 의해, 판단하는 단계; 상기 메모리 모듈에 저장된 상기 데이터 어레이를 이용하여 상기 HBM의 상기 PIM 회로에서 상기 연산 작업을 실행하도록 지시하는 메모리 모

듈 명령어 세트를, 상기 메모리 콘트롤러에 의해, 생성하는 단계; 및 상기 HBM에 저장된 상기 데이터 어레이를 이용하여 상기 HBM의 상기 PIM 회로에서 상기 연산 작업을 실행하도록 지시하는 HBM 명령어 세트를, 상기 메모리 콘트롤러에 의해, 생성하는 단계를 포함한다.

발명의 효과

[0009] 본 발명의 시스템은, 이종 메모리 시스템을 이용함에 따라 어플리케이션 실행 시간을 감소시키고 불필요한 용량의 확장을 피할 수 있다. 또한, 오프로드된 커널에 대하여 지정된 물리 주소 공간을 사용하는 메모리 모듈 명령어 세트와 HBM 명령어 세트를 생성함에 따라 별도의 주소 변환 과정을 필요로 하지 않는다. 이에 따라, 단일 메모리로 구성된 시스템 대비 용량 및 대역폭 측면에서 효율적이다.

도면의 간단한 설명

[0010] 도 1은 본 발명의 예시적인 실시예에 따른 이종 메모리 시스템을 포함하는 데이터 처리 시스템을 나타내는 블록도이다.

도 2는 도 1의 메모리 모듈(들)을 설명하는 도면이다.

도 3은 도 1의 HBM을 설명하는 도면이다.

도 4는 도 1의 메모리 콘트롤러를 설명하는 블록 다이어그램이다.

도 5는 도 1의 이종 메모리 시스템에서 사용되는 명령어 세트들을 설명하는 도면이다.

도 6 내지 도 8은 도 4의 메모리 콘트롤러의 동작을 설명하는 플로우 다이어그램들이다.

도 9는 본 발명의 실시예들에 따른 이종 메모리 시스템을 포함하는 시스템을 나타내는 블록 다이어그램이다.

도 10은 발명의 실시예들에 따른 이종 메모리 시스템이 적용된 데이터 센터를 나타낸 도면이다.

발명을 실시하기 위한 구체적인 내용

[0011] 도 1은 본 발명의 예시적인 실시예에 따른 이종 메모리 시스템을 포함하는 데이터 처리 시스템을 나타내는 블록도이다.

[0012] 도 1을 참조하면, 데이터 처리 시스템(100)은 심층 신경 망(deep neural networks)과 같은 러닝 시스템(learning systems)과 같은 어플리케이션들 또는 고성능 컴퓨팅(high-performance computing), 그래픽 동작 등과 같은 어플리케이션들을 실행하도록 구성될 수 있다. 이러한 어플리케이션들은 작업들(jobs) 또는 태스크들을 병렬 방식으로 협력하여 실행하고, 다른 데이터 세트들을 트레이닝하고, 높은 정확도로 학습하기 위하여 많은 연산 및 메모리 능력들을 필요로 하고, 전력 효율성 및 낮은 레이턴시를 중요시한다.

[0013] 데이터 처리 시스템(100)은 호스트 장치(110) 및 메모리 모듈(들)(200)과 HBM(300)으로 구성되는 이종 메모리 시스템(120)을 포함할 수 있다. 호스트 장치(110)는 전체적인 작업 또는 태스크가 많은 수의 컴퓨팅 엔티티들(예, 프로세서들, 프로세서들 내의 코어들, 및 메모리-내-처리(Processing-In-Memory: PIM) 회로(321))에서 병렬로 실행되는, 보다 작은 작업들로 분할되는 병렬 처리 접근법을 사용하여 전체적인 작업 또는 태스크를 해결하는 데 사용될 것이다. 태스크는 계층구조 등으로 구성되어 있는 다수의 작업들을 포함하고, 작업은 컴퓨팅 엔티티에 의해 실행되어야 하는 실행 가능 코드, 처리될 데이터, 컴퓨팅 엔티티에 의해 이종 메모리 시스템(120)으로부터 검색되고, 코드의 실행을 통해 조작되며, 이어서 저장될 데이터를 지칭할 수 있다.

[0014] 호스트 장치(110)는 프로세서(들)(112) 및 메모리 콘트롤러(114)를 포함할 수 있다. 하나의 실시예에서, 확장 로직 회로(116)는 메모리 콘트롤러(114) 내에 구비될 수 있다. 프로세서(들)(112)은 명령어들을 처리하고 관리하는 데이터 처리 시스템(100)의 주된 구성(primary component)으로, 운영 체제(operating system) 및 어플리케이션들의 실행을 주로 담당한다. 또한, 프로세서(들)(112)은 복잡한 작업 또는 태스크를 해결하기 위해 작업 부하가 병렬 처리 되도록 다수의 컴퓨팅 엔티티들에 분산될 수 있게 한다. 프로세서(들)(112)은 중앙 처리 유닛(Central Processing Unit: CPU), 디지털 시그널 프로세서(Digital Signal Processor: DSP), 그래픽 처리 유닛(Graphic Processing Unit: GPU), 암호화 처리 유닛(encryption processing unit), 물리 처리 유닛(physics processing unit), 머신 러닝 처리 유닛(machine learning processing unit) 등과 같은 처리 유닛을 포함할 수 있다.

[0015] 프로세서(들)(112)은 다양한 연산 작업들, 명령어들, 또는 커널들(kernels)의 실행을 다른 프로세서로 분산하거

나 이종 메모리 시스템(120)으로 오프로드하여 효율성을 향상시킬 수 있다. 커널은 함께 그룹화되어 작업 또는 정의 가능한 서브-작업(definable sub-task)을 실행하는 하나 또는 그 이상의 명령어들로서 정의된다. 프로세서(들)(112)에 의해 오프로드된 커널에 의해 HBM(300)의 PIM 회로(321)가 연산 처리를 수행하는 일 예가 설명될 것이다. 다양한 종류의 연산 처리 동작이 PIM 회로(321)에서 수행될 수 있으며, 일 예로서 인공 지능과 관련하여 뉴럴 네트워크 연산들 중 적어도 일부가 PIM 회로(321)에서 수행될 수 있다. 예컨대, 프로세서(들)(112)는 뉴럴 네트워크 연산들 중 적어도 일부가 PIM 회로(321)에 의해 수행될 수 있도록, 메모리 콘트롤러(114)를 통해 HBM(300)을 제어할 수 있을 것이다. 또한, 이하의 실시예에서는 확장 로직 회로(116)가 오프로드된 커널을 실행하기 위해 이종 메모리 시스템(120)을 제어하는 것으로 설명될 것이나, 본 발명의 실시예들은 이에 국한될 필요가 없다. 예컨대, 확장 로직 회로(116)는 메모리 콘트롤러(114) 내에 구비되는 구성에 해당하고, 메모리 콘트롤러(114)가 이종 메모리 시스템(120)을 제어하는 것으로 설명되어도 무방할 것이다.

[0016] 메모리 콘트롤러(114)는 확장 로직 회로(116)을 포함할 수 있다. 메모리 콘트롤러(114)는 하드웨어 모듈, 드라이버 모듈 및/또는 파일 시스템 모듈을 포함할 수 있다. 여기서, 모듈은 각각 하드웨어, 소프트웨어, 미들웨어 중 적어도 하나의 형태로 구현될 수 있다. 예컨대, 확장 로직 회로(116)는 메모리 콘트롤러(114)에 로드되는 소프트웨어 블록일 수 있다.

[0017] 확장 로직 회로(116)는 오프로드된 커널에 의해 HBM(300)의 PIM 회로(321)가 연산 처리를 수행할 때 연산 작업에 요구되는 데이터 어레이가 존재하는 메모리를 판단하는 보더 인덱스 값을 저장할 수 있다. 확장 로직 회로(116)는 보더 인덱스 값을 이용하여 데이터 어레이를 메모리 모듈(들)(200) 또는 HBM(300)으로부터 검색할 수 있다. 또한, 확장 로직 회로(116) 오프로드된 커널에 대해 별도의 주소 변환을 수행하지 않도록 하기 위해, 메모리 모듈(들)(200) 및 HBM(300) 각각에 지정된 물리 주소 공간을 사용하는 메모리 모듈 명령어 세트 및 HBM 명령어 세트를 생성할 수 있다.

[0018] 이종 메모리 시스템(120)은 메모리 모듈(들)(200)과 HBM(300)으로 구성되는 이종의 메모리 장치들을 포함할 수 있다. 메모리 모듈(들)(200)은 DDR(double data rate) 프로토콜을 통하여 호스트 장치(110)에 연결될 수 있다. DDR 프로토콜은 JEDEC(Joint Electron Device Engineering Council)의 메모리 표준 인터페이스 규격일 수 있다. 메모리 모듈(들)(200)은 DDR 인터페이스에 따라 호스트 장치(110)에 연결되지만, 본 발명은 여기에 제한되지 않을 것이다. 본 발명의 메모리 모듈(들)(200)은 DDR 인터페이스 이외의 다양한 종류의 통신 인터페이스를 통하여 호스트 장치(110)에 연결될 수 있다. 예를 들어, 통신 인터페이스는 ISA(Industry Standard Architecture), PCIe(Peripheral Component Interconnect Express), SATA(Serial Advanced Technology Attachment), SCSI(Small Computer System Interface), SAS(Serial Atached SCSI), UAS(USB(universal storage bus) Attached SCSI), iSCSI(internet Small Computer System Interface), Fiber Channel, FCoE(Fiber Channel over Ethernet) 등과 같은 것일 수 있다.

[0019] 메모리 모듈(들)(200)은 듀얼 인-라인 메모리 모듈(dual in-line memory module)로 구현될 수 있다. 메모리 모듈(들)(200)은 적어도 하나의 DRAM(dynamic random access memory)를 포함할 수 있다. 이하에서, 설명의 편의를 위하여, 메모리 모듈(들)(200)은 DIMM(200)으로 혼용될 수 있다.

[0020] HBM(300)은 JEDEC 표준의 HBM 프로토콜을 통하여 호스트 장치(110)에 연결될 수 있다. HBM 프로토콜은 3차원 적층 메모리들(예를 들어, DRAM)을 위한 고성능 랜덤 액세스 메모리(RAM) 인터페이스이다. HBM(300)은 일반적으로 다른 DRAM 기술들(예를 들어, DDR4, GDDR5 등)보다 실질적으로 더 작은 폼 팩터(form factor)에서, 더 적은 전력을 소비하면서, 더 넓은 대역폭을 달성한다. HBM(300)은 연산 능력을 제공하는 메모리-내-처리(Processing-In-Memory: PIM) 회로(321)를 포함할 수 있다.

[0021] 도 2는 도 1의 메모리 모듈(들)을 설명하는 도면이다.

[0022] 도 1 및 도 2를 참조하면, 메모리 모듈(들)(200)은 레지스터 클럭 드라이버(210, RCD) 및 메모리 장치들(220)을 포함할 수 있다. 메모리 장치들(220)은 DRAM 장치들일 수 있다. 그러나, 본 발명의 범위가 이에 한정되는 것은 아니며, 메모리 장치들(220)은 SDRAM (Synchronous DRAM), DDR SDRAM (Double Data Rate SDRAM), LPDDR SDRAM (Low Power Double Data Rate SDRAM), GDDR SDRAM (Graphics Double Data Rate SDRAM), DDR2 SDRAM, DDR3 SDRAM, DDR4 SDRAM, DDR5 SDRAM 등과 같은 휘발성 메모리 장치들 중 어느 하나일 수 있다. 이하, 설명의 편의를 위하여 메모리 장치들(220)은 DRAMs(220)로 통칭한다.

[0023] RCD(210)는 호스트 장치(110)로부터 커맨드/어드레스 및 데이터를 수신하고 DRAMs(220)에 클럭 신호 및 커맨드/어드레스 신호를 제공할 수 있다. 메모리 모듈(들)(200)은 임의의 유형의 메모리 모듈로 구현될 수 있다. 예컨

대, 메모리 모듈(들)(200)은 UDIMM(Unbuffered DIMM), RDIMM(Registered DIMM), LRDIMM(Load Reduced DIMM), FBDIMM(Fully Buffered DIMM), SODIMM(Small Outline DIMM) 등으로 구현될 수 있다.

[0024] 메모리 용량을 증가시키기 위해 마더보드에 실장되는 DIMM(200)의 수를 증가시킬 수 있다. 하지만, DIMM(200)은 마더보드에 장착되는 커넥터들의 수에 의해 16 또는 32 비트의 데이터 입출력을 위한 구성들로 제한될 수 있다.

[0025] 도 3은 도 1의 HBM을 설명하는 도면이다.

[0026] 도 1 및 도 3을 참조하면, HBM(300)은 서로 독립된 인터페이스를 갖는 다수의 채널들(CH1~CH8)을 포함함으로써 높은 대역폭(Bandwidth)을 가질 수 있다. HBM(300)은 다수개의 다이들을 포함할 수 있으며, 일 예로서 버퍼 다이(또는, 로직 다이(310))와 버퍼 다이(310) 위에 적층된 하나 이상의 코어 다이들(320)을 포함할 수 있다. 도 3의 예에서는, 제1 내지 제4 코어 다이들이 HBM(300)에 구비되는 예가 도시되었으나, 상기 코어 다이들(320)의 개수는 다양하게 변경될 수 있다. 코어 다이들(320)은 메모리 다이들로 지칭될 수 있다.

[0027] 코어 다이들(320) 각각은 하나 이상의 채널을 포함할 수 있다. 도 3에서는 코어 다이들(320) 각각이 두 개의 채널을 포함함에 따라 HBM(300)은 8 개의 채널들(CH1 ~ CH8)을 갖는 예가 도시된다. 예컨대, 제1 코어 다이는 제1 채널 및 제3 채널(CH1, CH3)을 포함하고, 제2 코어 다이는 제2 채널 및 제4 채널(CH2, CH4)을 포함하며, 제3 코어 다이는 제5 채널 및 제7 채널(CH5, CH7)을 포함하며, 제4 코어 다이는 제6 채널 및 제8 채널(CH6, CH8)을 포함할 수 있다.

[0028] 버퍼 다이(310)는 호스트 장치(110)와 통신하는 인터페이스 회로(311)를 포함할 수 있으며, 인터페이스 회로(311)를 통해 호스트 장치(110)로부터 커맨드/어드레스 및 데이터를 수신할 수 있다. 호스트 장치(110)는 채널에 대응하여 배치되는 버스들을 통해 커맨드/어드레스 및 데이터를 전송할 수 있으며, 채널 별로 버스가 구분되도록 형성되거나, 일부의 버스는 적어도 두 개의 채널들에 공유될 수도 있을 것이다. 인터페이스 회로(311)는 호스트 장치(110)가 메모리 동작 또는 연산 처리를 요청하는 채널로 커맨드/어드레스 및 데이터를 전달할 수 있다. 또한, 본 발명의 예시적인 실시예에 따라, 코어 다이들(320) 각각 또는 채널들 각각은 PIM 회로(321)를 포함할 수 있다.

[0029] 호스트 장치(110)는 다수의 연산 작업들 또는 커널들 중 적어도 일부가 HBM(300)에서 수행될 수 있도록 커맨드/어드레스 및 데이터를 제공할 수 있으며, 호스트 장치(110)가 지정하는 채널의 PIM 회로(321)에서 연산 처리가 수행될 수 있다. 일 예로서, 수신된 커맨드/어드레스가 연산 처리를 지시하는 경우, 해당 채널의 PIM 회로(321)는 호스트 장치(110)로부터의 데이터 및/또는 해당 채널에서 독출된 데이터를 이용한 연산 처리를 수행할 수 있다. 다른 예로서, HBM(300)의 해당 채널로 수신된 커맨드/어드레스가 메모리 동작을 지시하는 경우에는 데이터에 대한 억세스 동작이 수행될 수 있다.

[0030] 일 실시예에 따라, 채널들 각각은 다수 개의 뱅크들을 포함할 수 있고, 각각의 채널의 PIM 회로(321)에는 하나 이상의 프로세싱 소자들이 구비될 수 있다. 일 예로서, 각각의 채널에서 프로세싱 소자들의 개수는 뱅크들의 개수와 동일할 수 있으며, 또는 프로세싱 소자들의 개수가 뱅크들의 개수보다 적음에 따라 하나의 프로세싱 소자가 적어도 두 개의 뱅크들에 공유될 수도 있을 것이다. 각 채널의 PIM 회로(321)는 호스트 장치(110)에 의해 오프로드된 커널을 실행할 수 있다.

[0031] 한편, 버퍼 다이(310)는 스루 실리콘 비아(Through Silicon Via: TSV) 영역(312), HBM 물리 계층 인터페이스(PHYsical layer interface: HBM PHY) 영역(313) 및 직렬화기/역직렬화기(SERializer/DESerializer: SERDES) 영역(314)을 더 포함할 수 있다. TSV 영역(312)은 코어 다이들(320)과의 통신을 위한 TSV가 형성되는 영역이다. 각각의 채널(CH1~CH8)이 128 비트의 대역폭(bandwidth)을 갖는 경우, TSV들은 1024 비트의 데이터 입출력을 위한 구성들을 포함할 수 있다.

[0032] HBM PHY 영역(313)은 호스트 장치(110)와의 통신을 위해 다수의 입출력 회로를 포함할 수 있으며, 일 예로서 HBM PHY 영역(313)은 호스트 장치(110)와의 통신을 위한 하나 이상의 포트들을 포함할 수 있다. HBM PHY 영역(313)은 호스트 장치(110)와 HBM(300) 사이의 효율적인 통신에 요구되는 신호들, 주파수, 타이밍, 구동, 상세 동작 파라미터 및 기능성(functionality)을 위해 제공되는 물리적 또는 전기적 계층과 논리적 계층을 포함할 수 있다. HBM PHY 영역(313)은 메모리 셀에 대응하는 로우 및 칼럼을 선택하는 것, 메모리 셀에 데이터를 기입하는 것, 또는 기입된 데이터를 독출하는 것과 같은 메모리 인터페이싱을 수행할 수 있다. HBM PHY 영역(313)은 JEDEC 표준의 HBM 프로토콜의 특징들을 지원할 수 있다.

[0033] SERDES 영역(314)은 호스트 장치(110)의 프로세서(들)(112)의 프로세싱 스루풋이 증가함에 따라, 그리고 메모리

대역폭에 대한 요구들이 증가함에 따라, JEDEC(Joint Electron Device Engineering Council) 표준의 SERDES 인터페이스를 제공하는 영역이다. SERDES 영역(314)은 SERDES 송신기 부분, SERDES 수신기 부분 및 제어기 부분을 포함할 수 있다. SERDES 송신기 부분은 병렬-투-직렬 회로 및 송신기를 포함하고, 병렬 데이터 스트림을 수신하고, 수신된 병렬 데이터 스트림을 직렬화 할 수 있다. SERDES 수신기 부분은 수신기 증폭기, 등화기, 클럭 및 데이터 복원 회로 및 직렬-투-병렬 회로를 포함하고, 직렬 데이터 스트림을 수신하고, 수신된 직렬 데이터 스트림을 병렬화 할 수 있다. 제어기 부분은 에러 검출 회로, 에러 정정 회로 및 FIFO(First In First Out)와 같은 레지스터들을 포함할 수 있다.

[0034] 도 4는 도 1의 메모리 콘트롤러(114)를 설명하는 블록 다이어그램이다. 도 5는 도 1의 이종 메모리 시스템(120)에서 사용되는 명령어 세트들을 설명하는 도면이다.

[0035] 도 1 및 도 4를 참조하면, 메모리 콘트롤러(114)는 호스트 장치(110)의 장치 드라이버를 통해 오프로드된 커널이 이종 메모리 시스템(120)을 이용하여 커널 코드 및/또는 데이터와 관련되는 동작들을 수행하도록 구성되는 확장 로직 회로(116)를 포함할 수 있다. 장치 드라이버는 호스트 장치(110)의 운영 체계 및 하드웨어 장치 사이에 인터페이스를 정의(define) 및 설정하도록 운영 체계 내에 활용된다. 이하에서, 확장 로직 회로(116)는 오프로드된 커널을 실행하기 위한 하드웨어, 펌웨어, 소프트웨어 또는 이들의 결합 방식으로 구현되는 것을 통칭한다.

[0036] 확장 로직 회로(116)는 프로세서(들)(112)에 의해 오프로드된 커널이 이종 메모리 시스템(120)의 HBM(300)의 PIM 회로(321)에서 실행하도록 구성될 수 있다. 또한, 확장 로직 회로(116)는 오프로드된 커널의 작업 코드의 실행에 따라 처리될 데이터를 DIMM(200) 및/또는 HBM(300)으로부터 검색하고, 코드의 실행을 통해 조작된 데이터를 재정렬하여 DIMM(200) 및/또는 HBM(300)에 할당하도록 구성될 수 있다.

[0037] 오프로드된 커널은 데이터 어레이에 대해 수행될 수 있다. 데이터 어레이는 어레이와 같은 형태로 구성된 데이터 구조체를 말하며, 프로그래밍 언어에 있는 배열 형태의 데이터 구조 (예, array, set, map, deque 등)를 활용할 수 있다. 예컨대, 그래픽 프로세싱, 인공 지능(AI) 어플리케이션 등은 사용자의 데이터 접근이 특정 데이터에 편향적으로 쏠리는 경향이 있는데, 이러한 편향적인 선호는 일관성 및 지속성이 있어 추후 데이터 활용에 대한 활용 예측이 가능하다. 그리고, 확장 로직 회로(116)는 데이터 어레이의 활용 빈도에 따라 이종 메모리 시스템(120)의 재정렬을 실시할 수 있다. 예시적으로, 확장 로직 회로(116)는 높은 활용 빈도를 갖는 작은 용량의 데이터 어레이는 HBM(300)에 할당하고, 낮은 활용 빈도를 갖는 큰 용량의 데이터 어레이는 DIMM(200)에 할당하는 재정렬을 수행할 수 있다. 확장 로직 회로(116)에 의해 DIMM(200) 및/또는 HBM(300)에 데이터 어레이 형태로 연속적으로 할당됨에 따라 데이터 관리 및 주소 변환 과정이 간단해질 수 있다.

[0038] 확장 로직 회로(116)는 인덱스 테이블 로직 회로(402), 어드레스 맵핑 테이블 로직 회로(404) 및 커맨드 발생 회로(406)를 포함할 수 있다. 인덱스 테이블 로직 회로(402)는 데이터 어레이의 활용 빈도를 구분하는 보더 인덱스(Border index) 값을 저장할 수 있다. 일 예로, 보더 인덱스 값은 어떤 어플리케이션 (예, 그래픽 프로세싱, 인공 지능(AI) 어플리케이션)이 실행되는 동안 특정 데이터 어레이에 대한 실측 활용 회수(number of actual utilization)와 예측 활용 회수(number of predicted utilization)에 기초하여 설정될 수 있다. 보더 인덱스 값은 예측 활용 회수 대비 실측 활용 회수의 비율로 계산된 제1 값으로 설정될 수 있다. 다른 예로, 보더 인덱스 값은 HBM(300)의 대역폭과 DIMM(200)의 대역폭에 기초하여 설정될 수도 있다. 보더 인덱스 값은 DIMM(200)의 대역폭(예, 24GB/sec 정도) 대비 HBM(300)의 대역폭(예, 1024GB/sec 정도)의 비율로 계산된 제2 값(예, 4 내지 6 정도)으로 설정될 수 있다. 또 다른 예로, 보더 인덱스 값은 예측 활용 회수 대비 실측 활용 회수의 비율로 계산된 제1 값과 DIMM(200)의 대역폭 대비 HBM(300)의 대역폭의 비율로 계산된 제2 값이 동일해지는 조건을 만족하도록 설정될 수 있다.

[0039] 인덱스 테이블 로직 회로(402)는 오프로드된 커널에 대한 정보를 포함할 수 있다. 오프로드된 커널에 대한 정보는 오프로드된 커널과 관련된 데이터 어레이의 식별 번호와 식별 번호에 해당하는 데이터 어레이에 대한 베이스 어드레스, 인덱스 값 및/또는 데이터 사이즈를 포함할 수 있다. 베이스 어드레스는 해당 데이터 어레이가 저장되는 DIMM(200) 또는 HBM(300)의 첫번째 물리 주소를 나타내고, 인덱스 값은 베이스 어드레스와의 오프셋을 나타낸다. 인덱스 테이블 로직 회로(402)는 수신된 인덱스 값이 보더 인덱스 값 보다 작으면 해당 데이터 어레이가 HBM(300)에 존재한다고 판단하고, 크면 DIMM(200)에 존재한다고 판단할 수 있다. 그리고, 인덱스 테이블 로직 회로(402)는 베이스 어드레스와 인덱스 값에 기초하여 해당 데이터 어레이의 물리적 주소를 계산하는 어드레스 변환 동작이 수행되도록 구성될 수 있다.

[0040] 어드레스 맵핑 테이블 로직 회로(404)는 DIMM 맵핑 테이블(420) 및 HBM 맵핑 테이블(430)을 포함할 수 있다.

맵핑 테이블들은 식별 번호에 해당하는 데이터 어레이와 베이스 어드레스와의 상관 관계를 저장하는 레지스터들(또는 저장 요소들)로 구현될 수 있다. DIMM 맵핑 테이블(420)은 DIMM(200)에 대한 해당 데이터 어레이의 베이스 어드레스를 저장하고, HBM 맵핑 테이블(430)은 HBM(300)에 대한 해당 데이터 어레이의 베이스 어드레스를 저장할 수 있다.

[0041] 확장 로직 회로(116)는 인덱스 테이블 로직 회로(402) 및 어드레스 맵핑 테이블 로직 회로(404)를 이용하여 해당 데이터 어레이의 물리 주소 공간을 계산할 수 있다. 확장 로직 회로(116)는 보더 인덱스 값과 수신된 인덱스 값을 비교하는 인덱스 비교부를 이용하여 해당 데이터 어레이가 저장되어 있는 곳이 HBM(300) 인지 또는 DIMM(200) 인지를 판단할 수 있다.

[0042] 예시적으로, 확장 로직 회로(116)는 수신된 인덱스 값이 보더 인덱스 값 보다 작으면 해당 데이터 어레이가 HBM(300)에 저장되어 있다고 판단할 수 있다. 확장 로직 회로(116)는 HBM 어드레스 계산부(416)을 이용하여 HBM 맵핑 테이블(430)에 저장된 HBM(300)의 베이스 어드레스로부터 인덱스 값 만큼 떨어진 위치에 대응하는 제1 어드레스에 해당 데이터 어레이가 저장되는 것으로 판단할 수 있다. 즉, 제1 어드레스는 (베이스 어드레스+인덱스 값)으로 계산될 수 있다. 그리고, HBM 어드레스 계산부(416) 는 제1 어드레스로부터 데이터 사이즈를 곱한 제2 어드레스까지의 물리 주소 공간에 해당 데이터 어레이가 HBM(300)에 저장되어 있는 것으로 판단할 수 있다. 즉, 제2 어드레스는 (베이스 어드레스+인덱스 값)*데이터 사이즈)로 계산될 수 있다.

[0043] 예시적으로, 확장 로직 회로(116)는 수신된 인덱스 값이 보더 인덱스 값 보다 크면 해당 데이터 어레이가 DIMM(200)에 저장되어 있다고 판단할 수 있다. 확장 로직 회로(116)는 DIMM 어드레스 계산부(412)을 이용하여 DIMM 맵핑 테이블(420)의 DIMM(200)의 베이스 어드레스로부터 인덱스 값에서 보더 인덱스 값을 뺀 만큼 떨어진 위치에 대응하는 제3 어드레스에 데이터 어레이가 저장되는 것으로 판단할 수 있다. 즉, 제3 어드레스는 (베이스 어드레스+(인덱스 값-보더 인덱스 값))으로 계산될 수 있다. 그리고, DIMM 어드레스 계산부(412)는 제3 어드레스로부터 데이터 사이즈를 곱한 제4 어드레스까지의 물리 주소 공간에 해당 데이터 어레이가 DIMM(200)에 저장되어 있는 것으로 판단할 수 있다. 즉, 제4 어드레스는 ((베이스 어드레스+(인덱스 값- 보더 인덱스 값)*데이터 사이즈)로 계산될 수 있다.

[0044] 오프로드된 커널이 HBM(300)의 PIM 회로(321)에서 실행될 수 있다. PIM 회로(321)는 커널 코드에 따른 데이터 어레이가 HBM(300)에 저장되어 있는 경우 HBM(300)을 직접 억세스하여 데이터를 가져올 수 있다. 그런데, 커널 코드에 따른 데이터 어레이가 DIMM(200)에 저장되어 있는 경우, PIM 회로(321)는 DIMM(200)과 직접 연결된 인터페이스가 없기 때문에 DIMM(200)을 직접 억세스할 수 없다. 이에 따라, PIM 회로(321)는 메모리 콘트롤러(114)를 통해 DIMM(200)의 데이터를 제공받을 수 있다.

[0045] 확장 로직 회로(116)에 의해 해당 데이터 어레이가 HBM(300)에 저장되어 있다고 판단되면, 커맨드 발생 회로(406)는 HBM(300) 어드레스를 오퍼랜드(operand)로 사용할 수 있는 HBM(300)에 대한 명령어 세트를 생성할 수 있다. HBM(300)에 대한 명령어 세트(530)는, 도 5a에 도시된 바와 같이, 오피코드 파라미터, 벡터 사이즈 파라미터, 소스 파라미터, 및 목적지 파라미터를 포함할 수 있다.

[0046] 오피코드 파라미터는 오프로드된 커널의 명령어과 동일하게 표기될 수 있다. 명령어 종류에는 x86, RISC-V 등을 포함하고, 각 명령어 별로 오피코드 파라미터가 설정될 수 있다. 벡터 사이즈 파라미터는 호스트 장치(110) 내 DRAM 캐시 라인의 데이터 입출력 수와 데이터 어레이의 데이터 입출력 수와의 상관 관계를 나타낸다. 예컨대, DRAM 캐시 라인이 64B(byte)이고 데이터 어레이가 256B 인 경우, 벡터 사이즈 파라미터는 4로 설정될 수 있다. 즉, 64B DRAM 캐시 라인을 4개 가져오는 것과 동일하게 표기될 수 있다. 소스 파라미터는 데이터 어레이를 가져오는 소스 어드레스를 나타내고, HBM(300)의 어드레스로 표기될 수 있다. 목적지 파라미터는 HBM(300)의 PIM 회로(321)에서 연산 처리를 실행할 때 중간 결과 값을 저장하는 레지스터의 식별 번호를 표기할 수 있다. 예컨대, 레지스터는 메모리 콘트롤러(114)에 포함될 수 있고, 4번째 레지스터인 경우 4로 표기될 수 있다.

[0047] 확장 로직 회로(116)에 의해 해당 데이터 어레이가 DIMM(200)에 저장되어 있다고 판단되면, 커맨드 발생 회로(406)는 DIMM(200)에 대한 명령어 세트를 생성할 수 있다. DIMM(200)에 대한 명령어 세트(520)는, 도 5b에 도시된 바와 같이, 오피코드 파라미터, 벡터 사이즈 파라미터, 벡터 포지션 파라미터, 이미디어트 밸류 파라미터, 및 목적지 파라미터를 포함할 수 있다.

[0048] 오피코드 파라미터는 오프로드된 커널의 명령어과 동일하게 표기될 수 있다. 명령어 종류에는 x86, RISC-V 등을 포함하고, 각 명령어 별로 오피코드 파라미터가 설정될 수 있다. 벡터 사이즈 파라미터는 호스트 장치(110) 내 DRAM 캐시 라인의 데이터 입출력 수와 데이터 어레이의 데이터 입출력 수와의 상관 관계를 나타낸다. 예컨

대, DRAM 캐시 라인이 64B이고 데이터 어레이가 256B 인 경우, 벡터 사이즈 파라미터는 4로 설정될 수 있다. 즉, 64B DRAM 캐시 라인을 4개 가져오는 것과 동일하게 표기될 수 있다. 벡터 포지션 파라미터는 현재 명령어에서 활용하는 64B 데이터의 위치를 표기할 수 있다. 예컨대, 256B 데이터 어레이 중에서 64B-128B에 해당하는 2번째 데이터를 가져올 경우, 벡터 포지션 파라미터는 2로 표기될 수 있다. 이미디어트 밸류 파라미터는 DIMM(200)에서 가지고 온 캐시 라인 64B 만큼의 데이터 값 자체를 저장할 수 있다. 목적지 파라미터는 DIMM(200)에서 가지고 온 데이터를 이용하여 연산 처리를 실행할 때 중간 결과 값을 저장하는 레지스터의 식별 번호를 표기할 수 있다. 예컨대, 레지스터는 메모리 콘트롤러(114)에 포함될 수 있고, 4번째 레지스터인 경우 4로 표기될 수 있다.

[0049] 도 5a 및 도 5b에서 설명된 HBM(300)에 대한 명령어 세트(530)와 DIMM(200)에 대한 명령어 세트(520)에서, 목적지 파라미터와 관련된 레지스터 할당을 위하여, 확장 로직 회로(116)는 레지스터 할당기(414)를 더 포함할 수 있다. 레지스터 할당기(414)는 레지스터에 대한 점유를 확인할 수 있는 SRAM 테이블을 포함하는 로직 회로로 구현될 수 있다. 레지스터 할당기는 어떤 레지스터가 프로세싱 동작에 할당되고 어떤 레지스터가 현재 활용이 안되고 있는지를 표기할 수 있다.

[0050] 도 6 내지 도 8은 도 4의 메모리 콘트롤러의 동작을 설명하는 플로우 다이어그램들이다.

[0051] 도 1 내지 도 6을 참조하면, 단계 S610에서 호스트 장치(110)의 프로세서(들)(112)은 연산 작업의 효율성을 향상시키기 위하여, 커널의 실행을 다른 프로세서로 분산하거나 이종 메모리 시스템(120)으로 오프로드할 수 있다. 프로세서(들)(112)은 오프로드된 커널과 관련된 정보, 예컨대 데이터 어레이의 식별 번호와 식별 번호에 해당하는 데이터 어레이에 대한 베이스 어드레스, 인덱스 값 및/또는 데이터 사이즈를 장치 드라이버를 통해 메모리 콘트롤러(114)로 제공할 수 있다. 메모리 콘트롤러(114)는 프로세서(들)(112)에 의해 오프로드된 커널이 HBM(300)의 PIM 회로(321)에서 연산 처리 동작을 수행하도록 제어할 수 있다.

[0052] 단계 S620에서, 메모리 콘트롤러(114)는 인덱스 테이블 로직 회로(402)에 저장된 보더 인덱스 값을 읽을 수 있다. 보더 인덱스 값은 오프로드된 커널과 관련된 데이터 어레이가 존재하는 메모리를 판단하는 기준으로 설정될 수 있다. 본 실시예는 보더 인덱스 값이 오프로드된 커널과 관련된 데이터 어레이의 활용 빈도 비율 및 DIMM(200)과 HBM(300)의 대역폭 비율에 기초하여 설정되는 예가 설명되지만, 실시예에 따라, 오프로드된 커널과 관련된 다른 정보를 이용하여 설정될 수 있다.

[0053] 단계 S630에서, 메모리 콘트롤러(114)는 단계 S610에서 수신된 인덱스 값과 보더 인덱스 값을 비교하여 해당 데이터 어레이가 존재하는 메모리를 판단할 수 있다. 메모리 콘트롤러(114)는 수신된 인덱스 값이 보더 인덱스 값 보다 크면(YES) 단계 S650로 이동하고, 작으면(NO) 단계 S640로 이동할 수 있다. 단계 S640에서, 메모리 콘트롤러(114)는 해당 데이터 어레이가 HBM(300)에 존재한다고 판단하고 HBM(300)에 대한 메모리 동작을 수행할 수 있다. 단계 S650에서, 메모리 콘트롤러(114)는 해당 데이터 어레이가 DIMM(200)에 존재한다고 판단하고 DIMM(200)에 대한 메모리 동작을 수행할 수 있다.

[0054] 메모리 콘트롤러(114)는, 단계 S630의 비교 결과, 해당 데이터 어레이가 HBM(300)에 존재한다고 판단되면, 도 7에 도시된 바와 같이, 단계 S710 내지 단계 S750을 포함하는 단계 S640의 HBM(300)에 대한 메모리 동작을 수행할 수 있다. S710 단계에서, 메모리 콘트롤러(114)는 단계 S610에서 수신된 오프로드된 커널의 데이터 어레이에 대한 베이스 어드레스에 대응하는 HBM 맵핑 테이블(430)에 저장된 HBM(300)의 베이스 어드레스를 읽을 수 있다.

[0055] 단계 S720에서, 메모리 콘트롤러(114)는 데이터 어레이가 존재하는 HBM(300)의 물리 어드레스들을 계산할 수 있다. 메모리 콘트롤러(114)는 HBM(300)의 베이스 어드레스로부터 인덱스 값 만큼 떨어진 위치에 대응하는 제1 어드레스를 계산할 수 있다. 즉, 제1 어드레스는 (베이스 어드레스+인덱스 값)으로 계산될 수 있다. 그리고, 메모리 콘트롤러(114)는 제1 어드레스로부터 데이터 사이즈를 곱한 값에 대응하는 제2 어드레스를 계산할 수 있다. 즉, 제2 어드레스는 (베이스 어드레스+인덱스 값)*데이터 사이즈)로 계산될 수 있다. 메모리 콘트롤러(114)는 오프로드된 커널의 데이터 어레이가 HBM(300)의 제1 어드레스로부터 제2 어드레스까지의 물리 주소 공간에 저장되어 있는 것으로 판단할 수 있다.

[0056] 단계 S730에서, 메모리 콘트롤러(114)는 오프로드된 커널에 따라 연산 처리가 HBM(300)의 PIM 회로(321)에서 실행될 때 중간 결과 값을 저장하는 레지스터를 할당할 수 있다. 이것은 HBM 명령어 세트(530)에 포함되는 목적지 파라미터와 관련된다.

[0057] 단계 S740에서, 메모리 콘트롤러(114)는 HBM(300)에 대한 명령어 세트(530)를 생성할 수 있다. HBM(300)에 대한 명령어 세트(530)는, 도 5a에서 설명된 오피코드 파라미터, 벡터 사이즈 파라미터, 소스 파라미터, 및 목적

지 파라미터를 포함할 수 있다. 오피코드 파라미터는 오프로드된 커널의 명령어과 동일하게 표기되고, 벡터 사이즈 파라미터는 호스트 장치(110) 내 DRAM 캐시 라인의 데이터 입출력 수와 데이터 어레이의 데이터 입출력 수와의 상관 관계를 표시할 수 있다. 소스 파라미터는 데이터 어레이를 가져오는 HBM(300)의 어드레스로 표기되고, 목적지 파라미터는 단계 S730에서 할당된 레지스터의 식별 번호를 표기할 수 있다.

[0058] 단계 S750에서, 메모리 콘트롤러(114)는 단계 S740에서 생성된 HBM(300)에 대한 명령어 세트(530)를 HBM(300)의 PIM 회로(321)로 전송할 수 있다. 이 후, HBM(300)의 PIM 회로(321)는 HBM(300)에 대한 명령어 세트(530)에 따라 연산 처리 동작을 수행할 것이다(단계 S660).

[0059] 메모리 콘트롤러(114)는, 단계 S630의 비교 결과, 해당 데이터 어레이가 DIMM(200)에 존재한다고 판단되면, 도 8에 도시된 바와 같이, 단계 S810 내지 단계 S850을 포함하는 단계 S650의 DIMM(200)에 대한 메모리 동작을 수행할 수 있다. S810 단계에서, 메모리 콘트롤러(114)는 단계 S610에서 수신된 오프로드된 커널의 데이터 어레이에 대한 베이스 어드레스에 대응하는 DIMM 맵핑 테이블(420)에 저장된 DIMM(200)의 베이스 어드레스를 읽을 수 있다.

[0060] 단계 S820에서, 메모리 콘트롤러(114)는 데이터 어레이가 존재하는 DIMM(200)의 물리 어드레스들을 계산할 수 있다. 메모리 콘트롤러(114)는 DIMM 맵핑 테이블(420)의 DIMM(200)의 베이스 어드레스로부터 인덱스 값에서 보더 인덱스 값을 뺀 만큼 떨어진 위치에 대응하는 제3 어드레스를 계산할 수 있다. 즉, 제3 어드레스는 (베이스 어드레스+(인덱스 값-보더 인덱스 값))으로 계산될 수 있다. 그리고, 메모리 콘트롤러(114)는 제3 어드레스로부터 데이터 사이즈를 곱한 값에 대응하는 제4 어드레스를 계산할 수 있다. 즉, 제4 어드레스는 ((베이스 어드레스+(인덱스 값-보더 인덱스 값)*데이터 사이즈)로 계산될 수 있다. 메모리 콘트롤러(114)는 오프로드된 커널의 데이터 어레이가 DIMM(200)의 제3 어드레스로부터 제4 어드레스까지의 물리 주소 공간에 저장되어 있는 것으로 판단할 수 있다.

[0061] 단계 S825에서, 메모리 콘트롤러(114)는 단계 S820에서 계산된 DIMM(200)의 물리 어드레스들을 이용하여 DIMM(200)에 저장된 데이터 어레이를 독출할 수 있다. 이것은 HBM(300)의 PIM 회로(321)가 DIMM(200)과 직접 연결된 인터페이스가 없기 때문에 DIMM(200)을 직접 억세스할 수 없어서, 메모리 콘트롤러(114)를 통해 DIMM(200)의 데이터를 PIM 회로(321)로 제공하기 위함이다.

[0062] 단계 S830에서, 메모리 콘트롤러(114)는 오프로드된 커널에 따라 연산 처리가 HBM(300)의 PIM 회로(321)에서 실행될 때 중간 결과 값을 저장하는 레지스터를 할당할 수 있다. 이것은 DIMM 명령어 세트(520)에 포함되는 목적지 파라미터와 관련된다.

[0063] 단계 S840에서, 메모리 콘트롤러(114)는 DIMM(200)에 대한 명령어 세트(520)를 생성할 수 있다. DIMM(200)에 대한 명령어 세트(520)는, 도 5b에서 설명된 오피코드 파라미터, 벡터 사이즈 파라미터, 벡터 포지션 파라미터, 이미디어트 밸류 파라미터, 및 목적지 파라미터를 포함할 수 있다. 오피코드 파라미터는 오프로드된 커널의 명령어과 동일하게 표기되고, 벡터 사이즈 파라미터는 호스트 장치(110) 내 DRAM 캐시 라인의 데이터 입출력 수와 데이터 어레이의 데이터 입출력 수와의 상관 관계를 표시할 수 있다. 벡터 포지션 파라미터는 현재 명령어에서 활용하는 64B 데이터의 위치가 256B 데이터 어레이 중에서 몇번째 데이터 인지를 표기하고, 이미디어트 밸류 파라미터는 DIMM(200)에서 가지고 온 캐시 라인 64B 만큼의 데이터 값 자체를 나타낸다. 목적지 파라미터는 단계 S830에서 할당된 레지스터의 식별 번호를 표기할 수 있다.

[0064] 단계 S850에서, 메모리 콘트롤러(114)는 단계 S840에서 생성된 DIMM(200)에 대한 명령어 세트(520)를 HBM(300)의 PIM 회로(321)로 전송할 수 있다. 이 후, HBM(300)의 PIM 회로(321)는 DIMM(200)에 대한 명령어 세트(520)에 따라 연산 처리 동작을 수행할 것이다(S660)

[0065] 도 9는 본 발명의 실시예들에 따른 이종 메모리 시스템을 포함하는 시스템(1000)을 나타내는 블록 다이어그램이다.

[0066] 도 9를 참조하면, 시스템(1000)은 카메라(1100), 디스플레이(1200), 오디오 처리부(1300), 모뎀(1400), DRAM들(1500a, 1500b), 플래시 메모리들(1600a, 1600b), I/O 디바이스들(1700a, 1700b) 및 어플리케이션 프로세서(Application Processor, 1800, 이하 "AP"라고 칭함)를 포함할 수 있다. 시스템(1000)은 랩탑(laptop) 컴퓨터, 휴대용 단말기(mobile phone), 스마트폰(smart phone), 태블릿 PC(tablet personal computer), 웨어러블 기기, 헬스케어 기기, 또는 IOT(Internet Of Things) 기기로 구현될 수 있다. 또한, 시스템(1000)은 서버(Server), 또는 개인용 컴퓨터(Personal Computer)로 구현될 수도 있다.

[0067] 카메라(1100)는 사용자의 제어에 따라 정지 영상 또는 동영상을 촬영하고, 촬영한 이미지/영상 데이터를 저장하

거나 디스플레이(1200)로 전송할 수 있다. 오디오 처리부(1300)는 플래시 메모리 장치들(1600a, 1600b)나 네트워크의 컨텐츠에 포함된 오디오 데이터를 처리할 수 있다. 모뎀(1400)는 유/무선데이터 송수신을 위하여 신호를 변조하여 송신하고, 수신측에서 원래의 신호로 복구하기 위해 복조할 수 있다. I/O 디바이스들(1700a, 1700b)는 USB(Universal Serial Bus)나 스토리지, 디지털 카메라, SD(Secure Digital) 카드, DVD(Digital Versatile Disc), 네트워크 어댑터(Network adapter), 터치 스크린 등과 같은 디지털 입력 및/또는 출력 기능을 제공하는 기기들을 포함할 수 있다.

[0068] AP(1800)는 시스템(1000)의 전반적인 동작을 제어할 수 있다. AP(1800)는 플래시 메모리 장치들(1600a, 1600b)에 저장된 컨텐츠의 일부가 디스플레이(1200)에 표시되도록 디스플레이(1200)를 제어할 수 있다. AP(1800)는 I/O 디바이스들(1700a, 1700b)을 통하여 사용자 입력이 수신되면, 사용자 입력에 대응하는 제어 동작을 수행할 수 있다. AP(1800)는 AI(Artifitial Intelligence) 데이터 연산을 위한 전용 회로인 엑셀레이터(Accelerator) 블록을 포함하거나, AP(1800)와 별개로 엑셀레이터 칩(1820)을 구비할 수 있다. 엑셀레이터 블록 또는 엑셀레이터 칩(1820)에 추가적으로 DRAM(1500b)이 장착될 수 있다. 엑셀레이터는 AP(1800)의 특정 기능을 전문적으로 수행하는 기능 블록으로, 엑셀레이터는 그래픽 데이터 처리를 전문적으로 수행하는 기능 블럭인 GPU, AI 계산과 인퍼런스(Inference)를 전문적으로 수행하기 위한 블럭인 NPU(Neural Processing Unit), 데이터 전송을 전문적으로 하는 블록인 DPU(Data Processing Unit)를 포함할 수 있다

[0069] 시스템(1000)은 복수의 DRAM들(1500a, 1500b)을 포함할 수 있다. AP(1800)는 JEDEC(Joint Electron Device Engineering Council) 표준 규격에 맞는 커맨드와 모드 레지스터(MRS) 셋팅을 통하여 DRAM들(1500a, 1500b)을 제어하거나, 저전압/고속/신뢰성 등 업체 고유 기능 및 CRC(Cyclic Redundancy Check)/ECC(Error Correction Code) 기능을 사용하기 위하여 DRAM 인터페이스 규약을 설정하여 통신할 수 있다. 예를 들어 AP(1800)는 LPDDR4, LPDDR5 등의 JEDEC 표준 규격에 맞는 인터페이스로 DRAM(1500a)과 통신할 수 있으며, 엑셀레이터 블록 또는 엑셀레이터 칩(1820)는 DRAM(1500a)보다 높은 대역폭을 가지는 엑셀레이터용 DRAM(1500b)을 제어하기 위하여 새로운 DRAM 인터페이스 규약을 설정하여 통신할 수 있다.

[0070] 도 9에서는 DRAM들(1500a, 1500b)만을 도시하였으나, 이에 한정되지 않고 AP(1800)이나 엑셀레이터 칩(1820) 대역폭과 반응 속도, 전압 조건들을 만족한다면 PRAM이나 SRAM, MRAM, RRAM, FRAM 또는 Hybrid RAM의 메모리 등 어떤 메모리라도 사용 가능하다. DRAM들(1500a, 1500b)은 I/O 디바이스들(1700a, 1700b)이나 플래시 메모리들(1600a, 1600b) 보다 상대적으로 작은 레이턴시(latency)와 대역폭(bandwidth)를 가지고 있다. DRAM들(1500a, 1500b)은 시스템(1000)의 파워 온 시점에 초기화되고, 운영 체제와 어플리케이션 데이터가 로딩되어 운영 체제와 어플리케이션 데이터의 임시 저장 장소로 사용되거나 각종 소프트웨어 코드의 실행 공간으로 사용될 수 있다.

[0071] DRAM들(1500a, 1500b) 내에서는 더하기/빼기/곱하기/나누기 사칙 연산과 벡터 연산, 어드레스 연산, 또는 FFT(Fast Fourier Transform) 연산이 수행될 수 있다. 또한, DRAM들(1500a, 1500b) 내에서는 인퍼런스(inference)에 사용되는 수행을 위한 함수 기능(function)이 수행될 수 있다. 여기서, 인퍼런스는 인공 신경망(artificial neural network)을 이용한 딥러닝 알고리즘에서 수행될 수 있다. 딥러닝 알고리즘은 다양한 데이터를 통해 모델을 학습하는 트레이닝(training) 단계와 학습된 모델로 데이터를 인식하는 인퍼런스 단계를 포함할 수 있다. 실시예로서, 사용자가 카메라(1100)를 통해 촬영한 이미지는 신호 처리되어 DRAM(1500b) 내에 저장이 되며, 엑셀레이터 블록 또는 엑셀레이터 칩(1820)은 DRAM(1500b)에 저장된 데이터와 인퍼런스에 사용되는 함수를 이용하여 데이터를 인식하는 AI 데이터 연산을 수행할 수 있다.

[0072] 시스템(1000)에서, DRAM들(1500a, 1500b)은 도 1 내지 도 8에서 설명된 DIMM(200) 및 PIM 회로(321)를 포함하는 HBM(300)으로 구성되는 이종 메모리 시스템으로 구현될 수 있다. DRAM들(1500a, 1500b)은 AP(1800)에서 오프로드되는 커널에 따른 연산 작업을 수행할 수 있다. 메모리 콘트롤러(1810)는 HBM(300)의 PIM 회로(321)에서 연산 작업이 실행될 때, 연산 작업에 요구되는 데이터 어레이가 존재하는 메모리를 판단하는 기준이 되는 보더 인덱스 값을 이용하여 데이터 어레이를 DIMM(200) 또는 HBM(300)으로부터 검색할 수 있다. 또한, 메모리 콘트롤러(1810)는 오프로드된 커널에 대해 별도의 주소 변환을 수행하지 않도록 하기 위해, DIMM(200) 및 HBM(300) 각각에 지정된 물리 주소 공간을 사용하는 DIMM 명령어 세트 및 HBM 명령어 세트를 생성할 수 있다.

[0073] 시스템(1000)은 DRAM들(1500a, 1500b) 보다 큰 용량을 가진 복수의 스토리지 또는 복수의 플래시 메모리들(1600a, 1600b)을 포함할 수 있다. 엑셀레이터 블록 또는 엑셀레이터 칩(1820)은 플래시 메모리들(1600a, 1600b)를 이용하여 트레이닝(training) 단계와 AI 데이터 연산을 수행할 수 있다. 일 실시예로, 플래시 메모리들(1600a, 1600b)은 메모리 콘트롤러(1610) 내에 구비된 연산 장치를 사용하여 AP(1800) 및/내지 엑셀레이터 칩

(1820)이 수행하는 트레이닝(training) 단계과 인퍼런스 AI 데이터 연산을 보다 효율적으로 수행할 수 있다. 플래시 메모리들(1600a, 1600b)은 카메라(1100)를 통하여 찍은 사진을 저장하거나, 데이터 네트워크로 전송 받은 데이터를 저장할 수 있다. 예를 들어, 증강 현실(Augmented Reality)/가상 현실(Virtual Reality), HD(High Definition) 또는 UHD(Ultra High Definition) 컨텐츠를 저장할 수 있다.

[0074] 도 10은 발명의 일 실시예에 따른 이종 메모리 시스템이 적용된 데이터 센터를 나타낸 도면이다.

[0075] 도 10을 참조하면, 데이터 센터(3000)는 각종 데이터를 모아두고 서비스를 제공하는 시설로서, 데이터 스토리지 센터라고 지칭될 수도 있다. 데이터 센터(3000)는 검색 엔진 및 데이터 베이스 운용을 위한 시스템일 수 있으며, 은행 등의 기업 또는 정부기관에서 사용되는 컴퓨팅 시스템일 수 있다. 데이터 센터(3000)는 어플리케이션 서버들(3100 내지 3100n) 및 스토리지 서버들(3200 내지 3200m)을 포함할 수 있다. 어플리케이션 서버들(3100 내지 3100n)의 개수 및 스토리지 서버들(3200 내지 3200m)의 개수는 실시예에 따라 다양하게 선택될 수 있고, 어플리케이션 서버들(3100 내지 3100n)의 개수 및 스토리지 서버들(3200 내지 3200m)의 개수는 서로 다를 수 있다.

[0076] 어플리케이션 서버(3100) 또는 스토리지 서버(3200)는 프로세서(3110, 3210) 및 메모리(3120, 3220) 중 적어도 하나를 포함할 수 있다. 스토리지 서버(3200)를 예시로 설명하면, 프로세서(3210)는 스토리지 서버(3200)의 전반적인 동작을 제어할 수 있고, 메모리(3220)에 액세스하여 메모리(3220)에 로딩된 명령어 및/또는 데이터를 실행할 수 있다. 메모리(3220)는 DDR SDRAM(Double Data Rate Synchronous DRAM), HBM(High Bandwidth Memory), HMC(Hybrid Memory Cube), DIMM(Dual In-line Memory Module), Optane DIMM 또는 NVMDIMM(Non-Volatile DIMM)일 수 있다. 실시예에 따라, 스토리지 서버(3200)에 포함되는 프로세서(3210)의 개수 및 메모리(3220)의 개수는 다양하게 선택될 수 있다. 일 실시예에서, 프로세서(3210)와 메모리(3220)는 프로세서-메모리 페어를 제공할 수 있다. 일 실시예에서, 프로세서(3210)와 메모리(3220)의 개수는 서로 다를 수도 있다. 프로세서(3210)는 단일 코어 프로세서 또는 다중 코어 프로세서를 포함할 수 있다. 스토리지 서버(3200)에 대한 상기 설명은, 어플리케이션 서버(3100)에도 유사하게 적용될 수 있다. 실시예에 따라, 어플리케이션 서버(3100)는 스토리지 장치(3150)를 포함하지 않을 수도 있다. 스토리지 서버(3200)는 적어도 하나 이상의 스토리지 장치(3250)를 포함할 수 있다. 스토리지 서버(3200)에 포함되는 스토리지 장치(3250)의 개수는 실시예에 따라 다양하게 선택될 수 있다.

[0077] 어플리케이션 서버(3100) 또는 스토리지 서버(3200)에서, 메모리(3120, 3220)는 도 1 내지 도 8에서 설명된 DIMM(200) 및 PIM 회로(321)를 포함하는 HBM(300)으로 구성되는 이종 메모리 시스템으로 구현될 수 있다. 메모리(3120, 3220)는 프로세서(3110, 3210)에서 오프로드되는 커널에 따른 연산 작업을 수행하도록 제어하는 메모리 콘트롤러와 결합될 수 있다. 메모리 콘트롤러는 HBM(300)의 PIM 회로(321)에서 연산 작업이 실행될 때, 연산 작업에 요구되는 데이터 어레이가 존재하는 메모리를 판단하는 기준이 되는 보더 인덱스 값을 이용하여 데이터 어레이를 DIMM(200) 또는 HBM(300)으로부터 검색할 수 있다. 또한, 메모리 콘트롤러는 오프로드된 커널에 대해 별도의 주소 변환을 수행하지 않도록 하기 위해, DIMM(200) 및 HBM(300) 각각에 지정된 물리 주소 공간을 사용하는 DIMM 명령어 세트 및 HBM 명령어 세트를 생성할 수 있다.

[0078] 어플리케이션 서버들(3100 내지 3100n) 및 스토리지 서버들(3200 내지 3200m)은 네트워크(3300)를 통해 서로 통신할 수 있다. 네트워크(3300)는 FC(Fibre Channel) 또는 이더넷(Ethernet) 등을 이용하여 구현될 수 있다. 이 때, FC는 상대적으로 고속의 데이터 전송에 사용되는 매체이며, 고성능/고가용성을 제공하는 광 스위치를 사용할 수 있다. 네트워크(3300)의 액세스 방식에 따라 스토리지 서버들(3200 내지 3200m)은 파일 스토리지, 블록 스토리지, 또는 오브젝트 스토리지로서 제공될 수 있다.

[0079] 일 실시예에서, 네트워크(3300)는 SAN(Storage Area Network)와 같은 스토리지 전용 네트워크일 수 있다. 예를 들어, SAN은 FC 네트워크를 이용하고 FCP(FC Protocol)에 따라 구현된 FC-SAN일 수 있다. 다른 예를 들어, SAN은 TCP/IP 네트워크를 이용하고 iSCSI(SCSI over TCP/IP 또는 Internet SCSI) 프로토콜에 따라 구현된 IP-SAN일 수 있다. 다른 실시예에서, 네트워크(3300)는 TCP/IP 네트워크와 같은 일반 네트워크일 수 있다. 예를 들어, 네트워크(3300)는 FCoE(FC over Ethernet), NAS(Network Attached Storage), NVMe-oF(NVMe over Fabrics) 등의 프로토콜에 따라 구현될 수 있다.

[0080] 이하에서는, 어플리케이션 서버(3100) 및 스토리지 서버(3200)를 중심으로 설명하기로 한다. 어플리케이션 서버(3100)에 대한 설명은 다른 어플리케이션 서버(3100n)에도 적용될 수 있고, 스토리지 서버(3200)에 대한 설명은 다른 스토리지 서버(3200m)에도 적용될 수 있다.

[0081] 어플리케이션 서버(3100)는 사용자 또는 클라이언트가 저장 요청한 데이터를 네트워크(3300)를 통해 스토리지 서버들(3200 내지 3200m) 중 하나에 저장할 수 있다. 또한, 어플리케이션 서버(3100)는 사용자 또는 클라이언트가 독출 요청한 데이터를 스토리지 서버들(3200 내지 3200m) 중 하나로부터 네트워크(3300)를 통해 획득할 수 있다. 예를 들어, 어플리케이션 서버(3100)는 웹 서버 또는 DBMS(Database Management System) 등으로 구현될 수 있다.

[0082] 어플리케이션 서버(3100)는 네트워크(3300)를 통해 다른 어플리케이션 서버(3100n)에 포함된 메모리(3120n) 또는 스토리지 장치(3150n)에 액세스할 수 있고, 또는 네트워크(3300)를 통해 스토리지 서버(3200-3200m)에 포함된 메모리(3220-3220m) 또는 스토리지 장치(3250-3250m)에 액세스할 수 있다. 이로써, 어플리케이션 서버(3100)는 어플리케이션 서버들(3100-3100n) 및/또는 스토리지 서버들(3200-3200m)에 저장된 데이터에 대해 다양한 동작들을 수행할 수 있다. 예를 들어, 어플리케이션 서버(3100)는 어플리케이션 서버들(3100-3100n) 및/또는 스토리지 서버들(3200-3200m) 사이에서 데이터를 이동 또는 카피(copy)하기 위한 명령어를 실행할 수 있다. 이 때 데이터는 스토리지 서버들(3200-3200m)의 스토리지 장치(3250-3250m)로부터 스토리지 서버들(3200-3200m)의 메모리들(3220-3220m)을 거쳐서, 또는 바로 어플리케이션 서버들(3100-3100n)의 메모리(3120-3120n)로 이동될 수 있다. 네트워크(3300)를 통해 이동하는 데이터는 보안 또는 프라이버시를 위해 암호화된 데이터일 수 있다.

[0083] 스토리지 서버(3200)를 예시로 설명하면, 인터페이스(3254)는 프로세서(3210)와 콘트롤러(3251)의 물리적 연결 및 NIC(3240)와 콘트롤러(3251)의 물리적 연결을 제공할 수 있다. 예를 들어, 인터페이스(3254)는 스토리지 장치(3250)를 전용 케이블로 직접 접속하는 DAS(Direct Attached Storage) 방식으로 구현될 수 있다. 또한, 예를 들어, 인터페이스(3254)는 ATA(Advanced Technology Attachment), SATA(Serial ATA), e-SATA(external SATA), SCSI(Small Computer Small Interface), SAS(Serial Attached SCSI), PCI(Peripheral Component Interconnection), PCIe(PCI express), NVMe(NVM express), IEEE 1394, USB(universal serial bus), SD(secure digital) 카드, MMC(multi-media card), eMMC(embedded multi-media card), UFS(Universal Flash Storage), eUFS(embedded Universal Flash Storage), CF(compact flash) 카드 인터페이스 등과 같은 다양한 인터페이스 방식으로 구현될 수 있다.

[0084] 스토리지 서버(3200)는 스위치(3230) 및 NIC(3240)을 더 포함할 수 있다. 스위치(3230)는 프로세서(3210)의 제어에 따라 프로세서(3210)와 스토리지 장치(3250)를 선택적으로 연결시키거나, NIC(3240)과 스토리지 장치(3250)를 선택적으로 연결시킬 수 있다.

[0085] 일 실시예에서 NIC(3240)는 네트워크 인터페이스 카드, 네트워크 어댑터 등을 포함할 수 있다. NIC(3240)는 유선 인터페이스, 무선 인터페이스, 블루투스 인터페이스, 광학 인터페이스 등에 의해 네트워크(3300)에 연결될 수 있다. NIC(3240)는 내부 메모리, DSP, 호스트 버스 인터페이스 등을 포함할 수 있으며, 호스트 버스 인터페이스를 통해 프로세서(3210) 및/또는 스위치(3230) 등과 연결될 수 있다. 호스트 버스 인터페이스는, 앞서 설명한 인터페이스(3254)의 예시들 중 하나로 구현될 수도 있다. 일 실시예에서, NIC(3240)는 프로세서(3210), 스위치(3230), 스토리지 장치(3250) 중 적어도 하나와 통합될 수도 있다.

[0086] 어플리케이션 서버(3100-3100n) 또는 스토리지 서버(3200-3200m)에서 프로세서는 스토리지 장치(3150-3150n, 3250-3250m) 또는 메모리(3120-3120n, 3220-3220m)로 커맨드를 전송하여 데이터를 프로그램하거나 리드할 수 있다. 이 때 데이터는 ECC(Error Correction Code) 엔진을 통해 에러 정정된 데이터일 수 있다. 데이터는 데이터 버스 변환(Data Bus Inversion: DBI) 또는 데이터 마스킹(Data Masking: DM) 처리된 데이터로서, CRC(Cyclic Redundancy Code) 정보를 포함할 수 있다. 데이터는 보안 또는 프라이버시를 위해 암호화된 데이터일 수 있다.

[0087] 스토리지 장치(3150-3150n, 3250-3250m)는 프로세서로부터 수신된 리드 커맨드에 응답하여, 제어 신호 및 커맨드/어드레스 신호를 NAND 플래시 메모리 장치(3252-3252m)로 전송할 수 있다. 이에 따라 NAND 플래시 메모리 장치(3252-3252m)로부터 데이터를 독출하는 경우, RE(Read Enable) 신호는 데이터 출력 제어 신호로 입력되어, 데이터를 DQ 버스로 출력하는 역할을 할 수 있다. RE 신호를 이용하여 DQS(Data Strobe)를 생성할 수 있다. 커맨드와 어드레스 신호는 WE(Write Enable) 신호의 상승 엣지 또는 하강 엣지에 따라 페이지 버퍼에 래치될 수 있다.

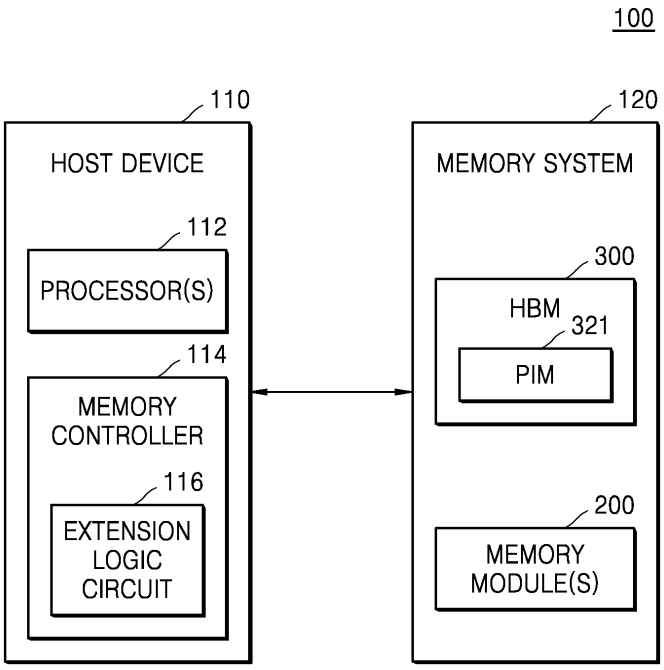
[0088] 콘트롤러(3251)는 스토리지 장치(3250)의 동작을 전반적으로 제어할 수 있다. 일 실시예에서, 콘트롤러(3251)는 SRAM(Static Random Access Memory)을 포함할 수 있다. 콘트롤러(3251)는 기입 커맨드에 응답하여 낸드 플래시(3252)에 데이터를 기입할 수 있고, 또는 독출 커맨드에 응답하여 낸드 플래시(3252)로부터 데이터를 독출

할 수 있다. 예를 들어, 기입 커맨드 및/또는 독출 커맨드는 스토리지 서버(3200) 내의 프로세서(3210), 다른 스토리지 서버(3200m) 내의 프로세서(3210m) 또는 어플리케이션 서버(3100, 3100n) 내의 프로세서(3110, 3110n)로부터 제공될 수 있다. DRAM(3253)은 낸드 플래시(3252)에 기입될 데이터 또는 낸드 플래시(3252)로부터 독출된 데이터를 임시 저장(버퍼링)할 수 있다. 또한, DRAM(3253)은 메타 데이터를 저장할 수 있다. 여기서, 메타 데이터는 사용자 데이터 또는 낸드 플래시(3252)를 관리하기 위해 콘트롤러(3251)에서 생성된 데이터이다. 스토리지 장치(3250)는 보안 또는 프라이버시를 위해 SE(Secure Element)를 포함할 수 있다.

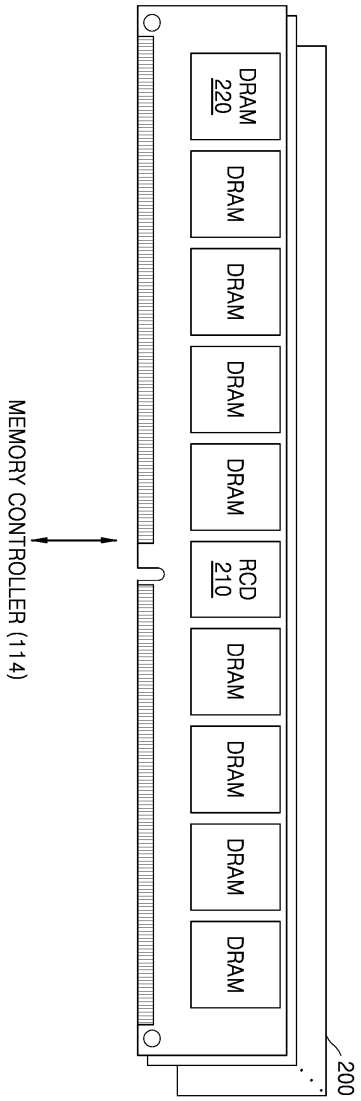
[0089] 본 발명은 도면에 도시된 제한된 수의 실시예들과 관련하여 설명되었으나 이는 예시적인 것에 불과하며, 본 기술 분야의 통상의 지식을 가진 자라면 이로부터 다양한 변경들 및 변형들, 그리고 균등한 다른 실시예들이 가능하다는 점을 인식할 것이다. 따라서, 첨부된 청구항들은 본 발명의 진정한 사상 및 범위 내에 속하는 바와 같은 모든 그러한 변경들 및 변형들을 커버하는 것을 의도한다.

도면

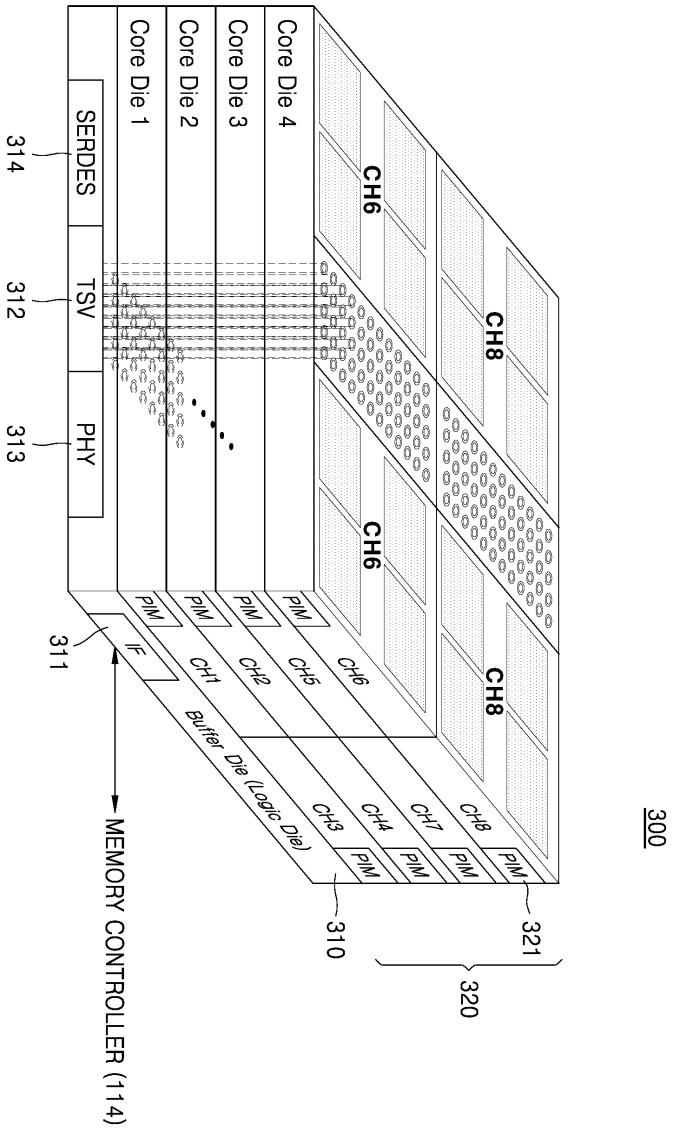
도면1



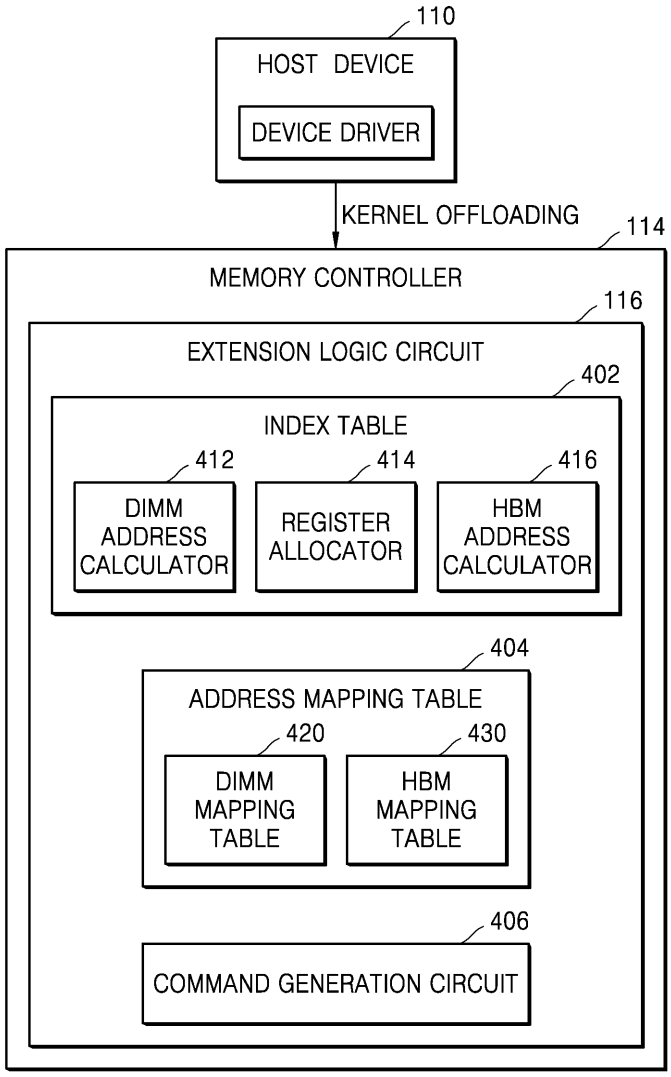
도면2


DRAM
220
DRAM
DRAM
DRAM
DRAM
RCD
210
DRAM
DRAM
DRAM
DRAM
200
MEMORY CONTROLLER (114)

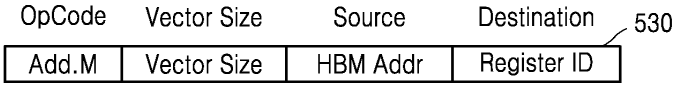
도면3


300
Core Die 4
Core Die 3
Core Die 2
Core Die 1
SERDES
TSV
PHY
314
312
313
311
CH6
CH8
CH6
CH8
PIM
CH5
CH6
CH1
CH2
CH3
CH4
CH7
CH8
Buffer Die (Logic Die)
IF
MEMORY CONTROLLER (114)
310
321
320

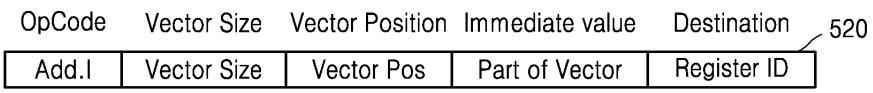
도면4



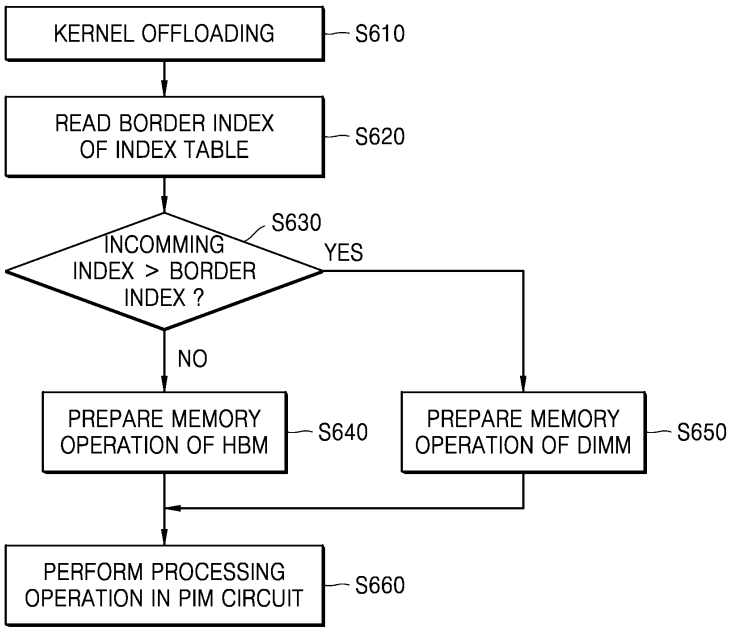
도면5a



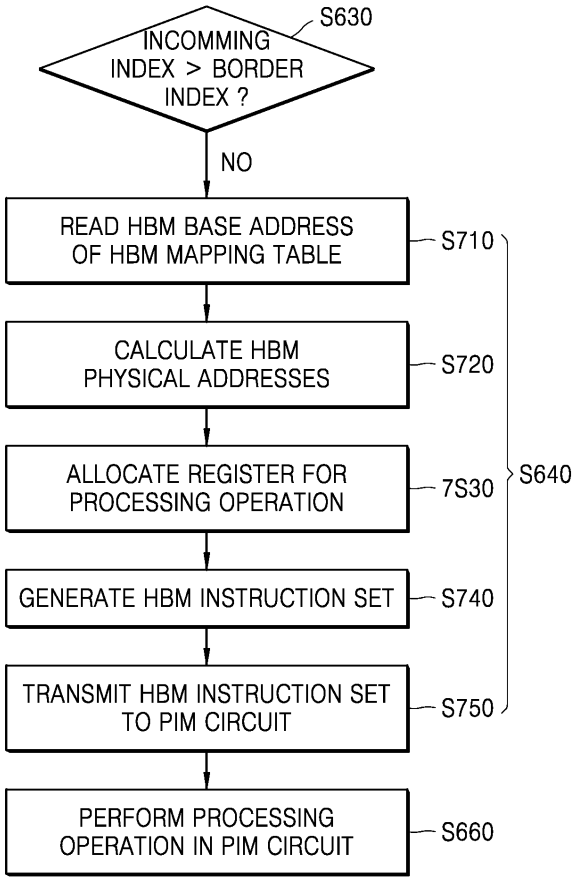
도면5b



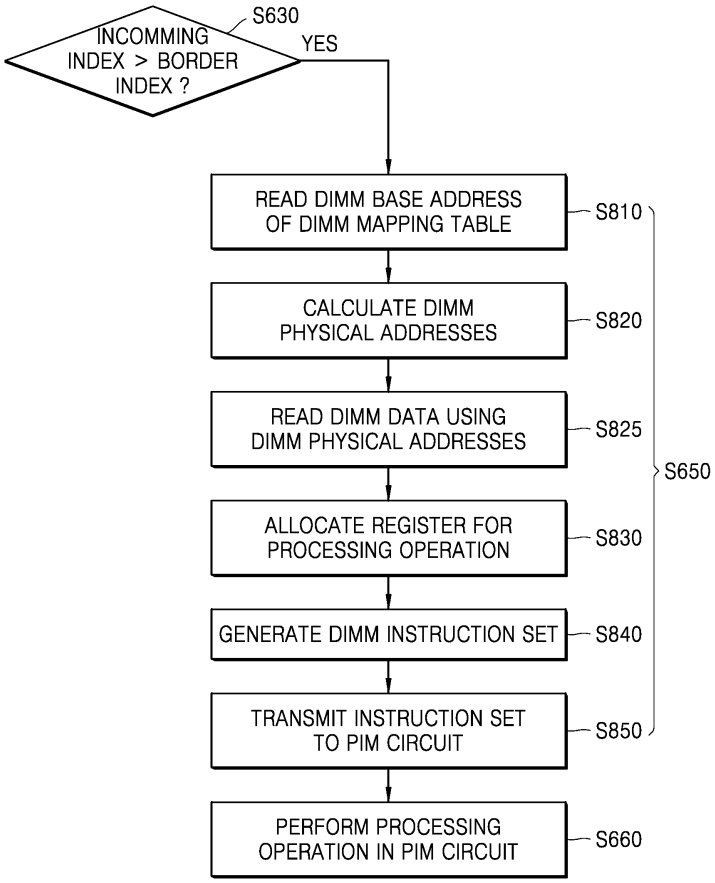
도면6


KERNEL OFFLOADING
S610
READ BORDER INDEX OF INDEX TABLE
S620
INCOMMING INDEX > BORDER INDEX ?
S630
YES
NO
PREPARE MEMORY OPERATION OF HBM
S640
PREPARE MEMORY OPERATION OF DIMM
S650
PERFORM PROCESSING OPERATION IN PIM CIRCUIT
S660

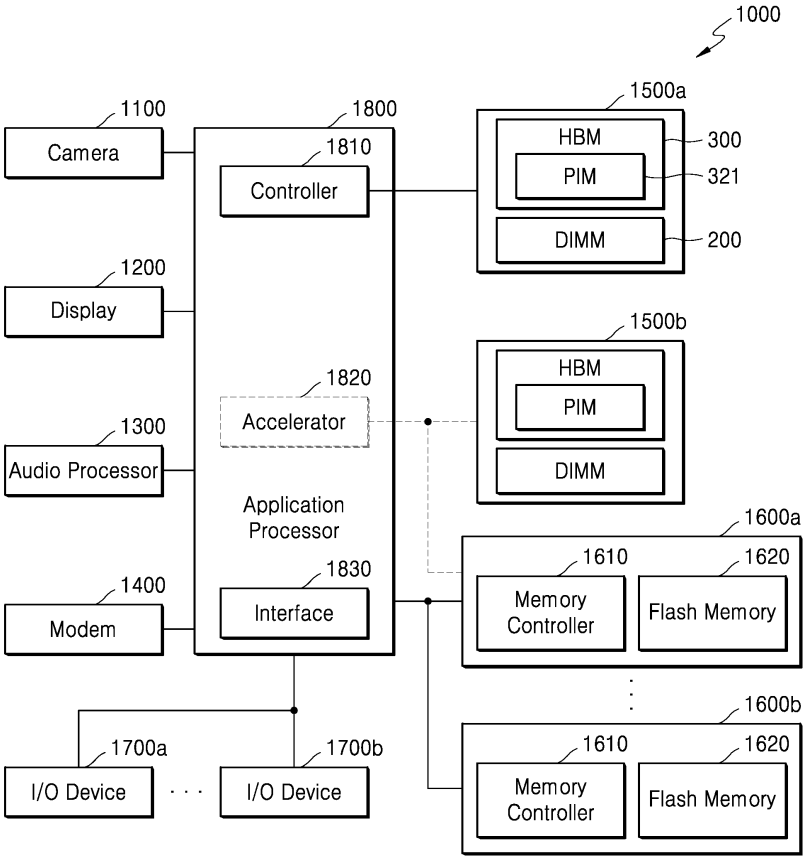
도면7


INCOMMING INDEX > BORDER INDEX ?
S630
NO
READ HBM BASE ADDRESS OF HBM MAPPING TABLE
S710
CALCULATE HBM PHYSICAL ADDRESSES
S720
ALLOCATE REGISTER FOR PROCESSING OPERATION
7S30
S640
GENERATE HBM INSTRUCTION SET
S740
TRANSMIT HBM INSTRUCTION SET TO PIM CIRCUIT
S750
PERFORM PROCESSING OPERATION IN PIM CIRCUIT
S660

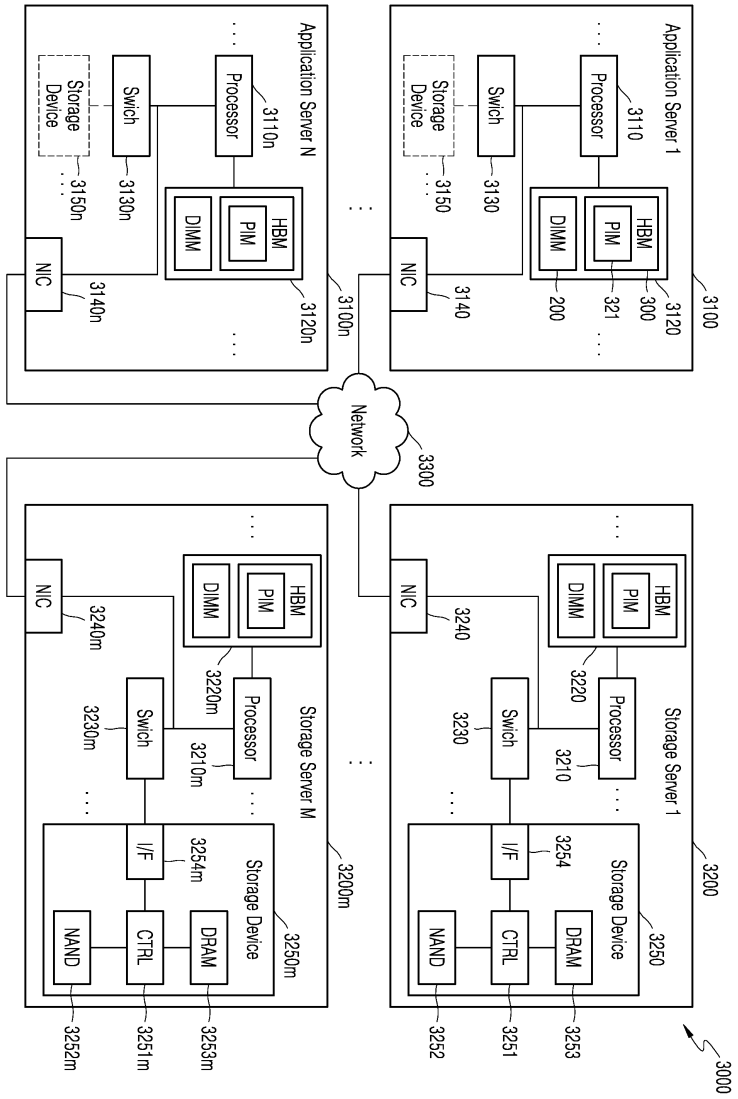
도면8


INCOMMING INDEX > BORDER INDEX ?
S630
YES
READ DIMM BASE ADDRESS OF DIMM MAPPING TABLE
S810
CALCULATE DIMM PHYSICAL ADDRESSES
S820
READ DIMM DATA USING DIMM PHYSICAL ADDRESSES
S825
S650
ALLOCATE REGISTER FOR PROCESSING OPERATION
S830
GENERATE DIMM INSTRUCTION SET
S840
TRANSMIT INSTRUCTION SET TO PIM CIRCUIT
S850
PERFORM PROCESSING OPERATION IN PIM CIRCUIT
S660

도면9


1000
1100
Camera
1800
1810
Controller
1500a
HBM
PIM
300
321
DIMM
200
1200
Display
1500b
HBM
PIM
DIMM
1820
Accelerator
1300
Audio Processor
Application Processor
1600a
1610
1620
Memory Controller
Flash Memory
1830
1400
Modem
Interface
1600b
1610
1620
Memory Controller
Flash Memory
1700a
I/O Device
1700b
I/O Device

도면10


Application Server 1
Processor
3110
Swich
3130
Storage Device
3150
NIC
3140
DIMM
PIM
HBM
200
321
300
3120
3100
Application Server N
3110n
3130n
3150n
3140n
3120n
3100n
Network
3300
Storage Server 1
3210
3230
3240
3220
3254
3250
I/F
CTRL
NAND
DRAM
3252
3251
3253
3200
3000
Storage Server M
3210m
3230m
3240m
3220m
3254m
3250m
3252m
3251m
3253m
3200m