



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2023-0169684
(43) 공개일자 2023년12월18일

- (51) 국제특허분류(Int. Cl.)
G06F 15/78 (2006.01) G06F 3/06 (2006.01)
G06F 9/26 (2017.01) G06F 9/30 (2018.01)
G06F 9/46 (2006.01)
- (52) CPC특허분류
G06F 15/7821 (2013.01)
G06F 3/0658 (2013.01)
- (21) 출원번호 10-2022-0070129
(22) 출원일자 2022년06월09일
심사청구일자 없음
- (71) 출원인
에스케이하이닉스 주식회사
경기도 이천시 부발읍 경충대로 2091
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암
동5가)
- (72) 발명자
김선욱
경기도 남양주시 도농로 34(다산동, 플루리움)
김창현
경기도 성남시 분당구 미금일로 136(구미동, 까치
마을건영빌라)
이원준
경기도 성남시 분당구 정자일로 80(정자동, 상록
마을보성아파트)
- (74) 대리인
이철희

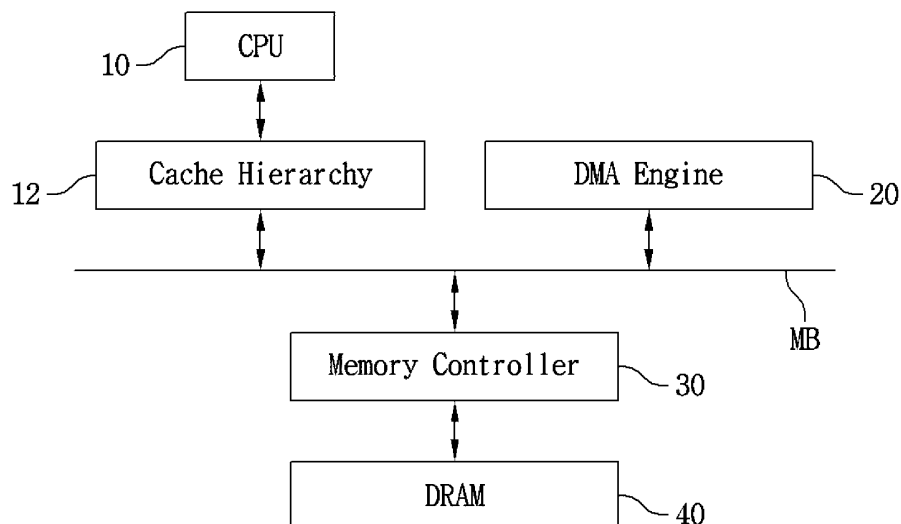
전체 청구항 수 : 총 22 항

(54) 발명의 명칭 프로세싱-인-메모리 컴퓨팅 시스템 및 그의 PIM 연산 오프로딩 방법

(57) 요약

본 발명은 PIM 컴퓨팅 시스템 및 그의 PIM 연산 오프로딩 방법을 개시하며, 상기 PIM 컴퓨팅 시스템은 DMA 엔진을 이용하여 PIM 연산 오프로딩을 수행하며 CPU의 메모리 리퀘스트에 대응하여 DMA 엔진이 DRAM에 저장된 디스크립터에 대하여 디스크립터 리퀘스트 및 PIM 리퀘스트를 수행함으로써 트랜잭션을 처리하도록 구성된다.

대표도 - 도1



(52) CPC특허분류

G06F 3/0659 (2013.01)

G06F 9/26 (2013.01)

G06F 9/3004 (2013.01)

G06F 9/466 (2013.01)

G06F 2213/28 (2013.01)

명세서

청구범위

청구항 1

프로세싱-인-메모리(Processing-in-Memory: 이하, PIM이라 함) 연산의 오프로딩을 위한 메모리 리퀘스트를 전송하는 중앙 처리 장치(Central Processing Unit: 이하, "CPU"라 함);

상기 PIM 연산을 위한 PIM 트랜잭션들을 디스크립터들로서 저장하는 메모리 어레이 및 상기 PIM 트랜잭션이 발생하면 상기 PIM 트랜잭션을 수행하는 PIM 디바이스를 포함하며, 상기 디스크립터 각각은 다음 디스크립터 주소(Next Descriptor Address), 오퍼랜드(Operand) 주소 정보 및 오퍼코드(Opcode)를 포함하는 디램(DRAM); 및

상기 메모리 리퀘스트에 의해 현재(Current) 디스크립터 주소와 테일(Tail) 디스크립터 주소를 저장하고, 첫번째의 상기 현재 디스크립터 주소와 마지막의 상기 테일 디스크립터 주소에 의해 순서대로 선택되는 상기 DRAM의 상기 디스크립터들 각각에 대한 디스크립터 리퀘스트와 PIM 리퀘스트를 순차적으로 제공하는 다이렉트 메모리 액세스(Direct Memory Access: 이하, "DMA"라 함) 엔진;을 구비하며,

상기 PIM 디바이스는,

상기 디스크립터 리퀘스트에 대응하여 상기 DMA 엔진이 상기 DRAM에서 리드하는 디스크립터 정보에서 상기 오퍼랜드 주소 정보와 상기 오퍼코드를 등록하고; 그리고,

상기 PIM 리퀘스트에 대응하여, 등록된 상기 오퍼랜드 주소 정보와 상기 오퍼코드를 이용하여 상기 PIM 트랜잭션을 수행함을 특징으로 하는 PIM 컴퓨팅 시스템.

청구항 2

제1 항에 있어서, 상기 DMA 엔진은,

상기 PIM 연산을 위한 순서에 해당하는 상기 디스크립터에 대하여 상기 디스크립터 리퀘스트를 제공하여 상기 디스크립터 정보를 리드하고;

상기 디스크립터 정보에서 상기 오퍼랜드 주소 정보와 상기 오퍼코드를 상기 PIM 디바이스에 등록하며; 그리고

리드된 상기 디스크립터 정보를 바탕으로 상기 PIM 트랜잭션을 발생시키는 상기 PIM 리퀘스트를 제공하는 PIM 컴퓨팅 시스템.

청구항 3

제1 항에 있어서,

상기 DMA 엔진은 이전 디스크립터의 상기 다음 디스크립터 주소에 의해 상기 디스크립터들을 순서대로 선택하는 PIM 컴퓨팅 시스템.

청구항 4

제1 항에 있어서, 상기 CPU는,

상기 PIM 연산 전에, 상기 PIM 연산을 위한 모든 PIM 트랜잭션들을 상기 DRAM의 상기 메모리 어레이에 상기 디스크립터들로서 저장한 후, 상기 메모리 리퀘스트를 상기 DMA 엔진에 전송하는 PIM 컴퓨팅 시스템.

청구항 5

제1 항에 있어서,

상기 디스크립터들 각각은 상기 다음 디스크립터 주소(Next Descriptor Address), 상기 오퍼랜드 주소 정보, 트랜스퍼 사이즈(Transfer Size) 및 트랜스퍼 상태(Transfer status)를 포함하는 데이터 구조를 가지며, 그리고,

상기 트랜스퍼 상태는 상기 오퍼코드를 포함하는 PIM 컴퓨팅 시스템.

청구항 6

제5 항에 있어서,

상기 오퍼랜드 주소 정보는 소스 주소(Source Address) 및 목적지 주소(Destination Address)를 포함하는 PIM 컴퓨팅 시스템.

청구항 7

제1 항에 있어서,

상기 DRAM은 메모리 컨트롤러를 통하여 연결된 메모리 버스를 통하여 상기 CPU 및 상기 DMA 엔진과 통신하며; 그리고,

상기 메모리 컨트롤러는 상기 DRAM에 대한 상기 메모리 버스를 통한 상기 디스크립터 리퀘스트, 상기 PIM 리퀘스트 및 데이터 입출력을 스케줄링하는 PIM 컴퓨팅 시스템.

청구항 8

제1 항에 있어서,

상기 PIM 디바이스는 오퍼랜드 레지스터, 구성(Configuration) 레지스터 및 디스크립터 레지스터를 포함하는 PIM 컨트롤 레지스터 세트를 포함하며;

상기 디스크립터 레지스터에는 상기 PIM 연산이 시작되기 전에 상기 CPU에서 제공되는 디스크립터 주소 정보들이 등록되며;

상기 PIM 디바이스는 상기 디스크립터 리퀘스트를 위하여 상기 DMA 엔진에서 선택된 디스크립터 주소를 상기 디스크립터 레지스터에 등록된 상기 디스크립터 주소 정보와 비교하여 상기 디스크립터 리퀘스트를 식별하며; 그리고,

상기 PIM 디바이스는 식별된 상기 디스크립터 리퀘스트에 대응하여 상기 DMA 엔진이 리드하여 제공하는 상기 디스크립터 주소에 해당하는 상기 오퍼랜드 주소 정보와 상기 오프 코드를 수신하며, 상기 오퍼랜드 주소 정보를 상기 오퍼랜드 레지스터에 등록하고 상기 오프코드를 상기 구성 레지스터에 등록하는 PIM 컴퓨팅 시스템.

청구항 9

제8 항에 있어서, 상기 PIM 디바이스는,

상기 DMA 엔진이 상기 PIM 트랜잭션을 발생시키는 상기 PIM 리퀘스트를 제공하면, 상기 PIM 리퀘스트를 위하여 상기 DMA 엔진에서 제공된 오퍼랜드 주소 정보를 상기 오퍼랜드 레지스터에 등록된 상기 오퍼랜드 주소 정보와 비교하여 상기 PIM 리퀘스트를 식별하고; 그리고,

식별된 상기 PIM 리퀘스트에 대응하여 상기 구성 레지스터에 등록된 상기 오프코드에 따라 상기 PIM 트랜잭션을 실행하는 PIM 컴퓨팅 시스템.

청구항 10

제9 항에 있어서,

상기 PIM 디바이스는 상기 PIM 트랜잭션이 완료되면 상기 오퍼랜드 레지스터 및 상기 구성 레지스터를 클리어하는 PIM 컴퓨팅 시스템.

청구항 11

제1 항에 있어서,

상기 DMA 엔진은 상기 PIM 트랜잭션이 완료되면 라이트(Write) 리퀘스트를 제공하여 상기 PIM 트랜잭션이 완료된 상기 디스크립터에 완료 비트(Complete Bit)를 라이트하는 PIM 컴퓨팅 시스템.

청구항 12

제1 항에 있어서,

상기 DMA 엔진은 마지막의 상기 테일 디스크립터에 대한 상기 PIM 트랜잭션이 완료(Complete)되면 상기 CPU에 인터럽트(Interrupt) 신호를 제공하는 PIM 컴퓨팅 시스템.

청구항 13

제1 항에 있어서, 상기 PIM 디바이스는,

상기 오퍼랜드 주소 정보의 등록을 위한 오퍼랜드 레지스터, 상기 오퍼코드의 등록을 위한 구성(Configuration) 레지스터 및 디스크립터 주소 정보들이 등록된 디스크립터 레지스터를 포함하는 PIM 컨트롤 레지스터 세트를 포함하며, 주소 정보 및 명령 정보를 수신하고, 연산 제어 신호를 제공하는 PIM 인터페이스 유닛; 및

상기 연산 제어 신호에 대응하여 상기 PIM 트랜잭션을 위한 상기 디스크립터 리퀘스트 및 상기 PIM 리퀘스트에 대응한 연산을 수행하고, 상기 PIM 리퀘스트에 의한 연산 결과를 제공하는 PIM 엔진;를 포함하며,

상기 PIM 인터페이스 유닛은,

수신된 상기 주소 정보를 상기 오퍼랜드 주소 정보와 상기 디스크립터 주소 정보들과 비교함으로써 상기 디스크립터 리퀘스트와 상기 PIM 리퀘스트의 상기 명령 정보를 식별하고;

상기 디스크립터 리퀘스트로 식별되면, 상기 DAM 엔진이 리드한 상기 디스크립터 정보에서 상기 오퍼랜드 주소 정보를 상기 오퍼랜드 레지스터에 등록하고 상기 오퍼코드를 상기 구성 레지스터에 등록하며;

상기 PIM 리퀘스트로 식별되면, 상기 구성 레지스터에 등록된 상기 오퍼랜드 주소 정보 및 상기 오퍼코드를 이용하여 상기 PIM 트랜잭션의 실행을 위한 상기 연산 제어 신호를 제공하는 PIM 컴퓨팅 시스템.

청구항 14

제13 항에 있어서,

상기 PIM 인터페이스 유닛은 상기 PIM 리퀘스트로 식별된 경우 상기 PIM 엔진과 상기 메모리 어레이의 데이터 경로를 형성하는 스위칭 회로를 더 포함하는 PIM 컴퓨팅 시스템.

청구항 15

CPU가 PIM 연산의 오프로딩을 위하여 메모리 리퀘스트를 전송하여서 DMA 엔진에 현재(Current) 디스크립터 주소와 테일(Tail) 디스크립터 주소를 저장하는 단계;

상기 PIM 연산 전, 상기 CPU가 상기 PIM 연산을 위한 모든 PIM 트랜잭션들을 DRAM의 메모리 어레이에 디스크립터들로서 저장하고 상기 DRAM의 PIM 디바이스에 디스크립터 주소 정보들을 등록하는 단계;

상기 DMA 엔진이 상기 PIM 연산을 위하여 상기 현재 디스크립터 주소부터 마지막의 상기 테일 디스크립터 주소까지 순서대로 디스크립터를 선택하고, 선택된 디스크립터에 대한 디스크립터 리퀘스트를 제공하여 디스크립터 정보를 리드하는 단계;

상기 DMA 엔진이 리드된 상기 디스크립터 정보에서 상기 오퍼랜드 주소 정보와 상기 오프 코드를 상기 PIM 디바이스에 제공하는 단계;

상기 PIM 디바이스가 상기 디스크립터 리퀘스트를 식별하고 상기 DMA 엔진에서 제공되는 상기 오퍼랜드 주소 정보와 상기 오퍼코드를 등록하는 단계;

상기 DMA 엔진이 리드된 상기 디스크립터 정보를 바탕으로 상기 PIM 트랜잭션을 발생시키는 상기 PIM 리퀘스트를 제공하는 단계;

상기 PIM 디바이스가 상기 PIM 리퀘스트를 식별하고 등록된 상기 오퍼랜드 주소 정보와 상기 오퍼코드를 이용하여 상기 PIM 트랜잭션을 수행하는 단계; 및

상기 현재 디스크립터 주소부터 마지막의 상기 테일 디스크립터 주소까지 상기 디스크립터들에 대하여 상기 디스크립터 리퀘스트와 상기 PIM 리퀘스트의 제공을 수행하여 상기 PIM 연산을 완료하는 단계;를 포함함을 특징으로 하는 PIM 컴퓨팅 시스템의 PIM 연산 오프로딩 방법.

청구항 16

제15 항에 있어서,

상기 메모리 어레이의 상기 디스크립터들 각각은 다음 디스크립터 주소(Next Descriptor Address), 상기 오퍼랜드 주소 정보, 트랜스퍼 사이즈(Transfer Size) 및 트랜스퍼 상태(Transfer status)를 포함하는 데이터 구조를 갖도록 설정되며, 그리고, 상기 트랜스퍼 상태는 상기 오퍼코드를 포함하도록 설정되는 PIM 컴퓨팅 시스템의 PIM 연산 오프로딩 방법.

청구항 17

제16 항에 있어서,

상기 디스크립터 리퀘스트를 제공하기 위한 상기 디스크립터의 선택은 이전 디스크립터의 상기 다음 디스크립터 주소에 의해 수행되는 PIM 컴퓨팅 시스템의 PIM 연산 오프로딩 방법.

청구항 18

제15 항에 있어서,

오퍼랜드 레지스터, 구성(Configuration) 레지스터 및 디스크립터 레지스터를 포함하는 PIM 컨트롤 레지스터 세트가 상기 PIM 디바이스에 세팅되며;

상기 PIM 연산이 시작되기 전에 상기 CPU에서 제공되는 상기 디스크립터 주소 정보들이 상기 디스크립터 레지스터에 등록되고; 그리고,

상기 디스크립터 정보가 리드될 때, 상기 오퍼랜드 주소 정보가 상기 오퍼랜드 레지스터에 등록되며, 상기 오퍼코드가 상기 구성 레지스터에 등록되는 PIM 컴퓨팅 시스템의 PIM 연산 오프로딩 방법.

청구항 19

제15 항에 있어서,

상기 PIM 디바이스는 상기 디스크립터 리퀘스트를 위하여 상기 DMA 엔진에서 선택된 디스크립터 주소와 등록된 상기 디스크립터 주소 정보를 비교함으로써 상기 PIM 디바이스가 디스크립터 리퀘스트를 식별하는 PIM 컴퓨팅 시스템의 PIM 연산 오프로딩 방법.

청구항 20

제15 항에 있어서,

상기 PIM 디바이스는 상기 PIM 리퀘스트를 위하여 상기 DMA 엔진에서 제공된 오퍼랜드 주소 정보와 등록된 상기 오퍼랜드 주소 정보를 비교함으로써 상기 PIM 리퀘스트를 식별하는 PIM 컴퓨팅 시스템의 PIM 연산 오프로딩 방법.

청구항 21

제20 항에 있어서,

상기 PIM 디바이스는 식별된 상기 PIM 리퀘스트에 대응하여 등록된 상기 오퍼코드에 따라 상기 PIM 트랜잭션을 실행하는 PIM 컴퓨팅 시스템의 PIM 연산 오프로딩 방법.

청구항 22

제15 항에 있어서,

상기 PIM 트랜잭션이 완료되면 등록된 상기 오퍼랜드 주소 정보와 상기 오퍼코드가 클리어되는 PIM 컴퓨팅 시스템의 PIM 연산 오프로딩 방법.

발명의 설명

기술 분야

[0001] 본 발명은 프로세싱-인-메모리(Processing-in-Memory: 이하, PIM이라 함) 컴퓨팅 시스템에 관한 것으로서, 보다 상세하게는 다이렉트 메모리 액세스(Direct Memory Access: 이하, DMA라 함) 엔진을 이용하여 PIM 연산 오프로딩을 수행하는 PIM 컴퓨팅 시스템 및 그의 PIM 연산 오프로딩 방법에 관한 것이다.

배경 기술

- [0002] 반도체 메모리를 이용하는 시스템의 중앙 처리 장치(Central Processing Unit: 이하, CPU라 함)는 반도체 메모리에 저장된 코드나 데이터를 읽고(read), 지정된 프로그램을 실행(execute)하며, 그 결과를 반도체 메모리에 저장할 수 있다. 반도체 메모리의 일례로서, 디램(DRAM)이 예시될 수 있다.
- [0003] 최근 반도체 메모리는 PIM 연산 기능을 갖도록 개발되고 있다. PIM 연산 기능은 머신 러닝(Machine Learning), 고성능 컴퓨팅 및 빅 데이터의 처리 분야 등에서 데이터 이동(Move)시 발생하는 정체를 해결하기 위하여 반도체 메모리 내부에서 연산을 수행하는 기능을 의미한다.
- [0004] 상기와 같이, PIM 연산 기능을 갖는 반도체 메모리를 이용하는 시스템은 PIM 컴퓨팅 시스템이라 할 수 있다.
- [0005] 일반적인 경우, PIM 연산은 스탠다드 메모리 리퀘스트(Standard Memory Request)를 사용하여 수행될 수 있고, PIM 컴퓨팅 시스템은 상기한 PIM 연산을 제어하기 위한 DMA 엔진을 구비할 수 있다.
- [0006] PIM 컴퓨팅 시스템의 경우, CPU는 PIM 연산을 위하여 대용량의 메모리 리퀘스트를 DRAM으로 전달하며, DMA 엔진은 메모리 리퀘스트에 대응하여 PIM 연산을 제어할 수 있다.
- [0007] PIM 컴퓨팅 시스템에서, CPU와 DRAM 사이에는 캐시 계층(Cache Hierarchy)이 존재한다. 그러므로, CPU에서 DRAM으로 전달되는 메모리 리퀘스트는 캐시 계층으로 인한 제한에 의하여 데이터 일관성(Data Consistency)과 관련된 오버헤드(Overhead)를 발생시킬 수 있다.
- [0008] 일반적으로 상기한 오버헤드가 발생하는 문제점은 PIM 컴퓨팅 시스템의 설계 변경을 통하여 해결될 수 있다. PIM 컴퓨팅 시스템의 설계 변경은 DMA 엔진의 설계 변경을 요구할 수 있다.
- [0009] 그러나, 상기와 같은 PIM 컴퓨팅 시스템 또는 DMA의 설계 변경은 구현이 복잡하고 범용성이 제한되는 문제점을 가질 수 있다.
- [0010] 따라서, PIM 컴퓨팅 시스템은 PIM 컴퓨팅 시스템 또는 DMA 엔진의 설계를 변경하지 않고 대용량의 메모리 리퀘스트에 대응한 PIM 연산을 수행할 수 있도록 개발될 필요가 있다.

발명의 내용

해결하려는 과제

- [0011] 본 발명의 목적은 DMA 엔진을 이용하여 캐시 계층(Cache Hierarchy)에 관련된 오버헤드 이슈없이 PIM 연산을 위한 대용량의 메모리 리퀘스트를 처리할 수 있고, PIM 연산을 효과적으로 수행할 수 있는 PIM 컴퓨팅 시스템 및 그의 PIM 연산 오프로딩 방법을 제공함에 있다.
- [0012] 본 발명의 다른 목적은 DMA 엔진의 설계를 변경하지 않고 대용량의 메모리 리퀘스트에 대응한 PIM 연산을 DMA 엔진에 오프로딩할 수 있는 PIM 컴퓨팅 시스템 및 그의 PIM 연산 오프로딩 방법을 제공함에 있다.

과제의 해결 수단

- [0013] 본 발명의 PIM 컴퓨팅 시스템은, PIM 연산의 오프로딩을 위한 메모리 리퀘스트를 전송하는 CPU; 상기 PIM 연산을 위한 PIM 트랜잭션들을 디스크립터들로서 저장하는 메모리 어레이 및 상기 PIM 트랜잭션이 발생하면 상기 PIM 트랜잭션을 수행하는 PIM 디바이스를 포함하며, 상기 디스크립터 각각은 다음 디스크립터 주소(Next Descriptor Address), 오퍼랜드(Operand) 주소 정보 및 오퍼코드(Opcode)를 포함하는 디램(DRAM); 및 상기 메모리 리퀘스트에 의해 현재(Current) 디스크립터 주소와 테일(Tail) 디스크립터 주소를 저장하고, 첫번째의 상기 현재 디스크립터 주소와 마지막의 상기 테일 디스크립터 주소에 의해 순서대로 선택되는 상기 DRAM의 상기 디스크립터들 각각에 대한 디스크립터 리퀘스트와 PIM 리퀘스트를 순차적으로 제공하는 DMA 엔진;을 구비하며, 상기 PIM 디바이스는, 상기 디스크립터 리퀘스트에 대응하여, 상기 DMA 엔진으로 리드되는 디스크립터 정보에서 상기 오퍼랜드 주소 정보와 상기 오퍼코드를 상기 PIM 디바이스에 등록하고; 그리고, 상기 PIM 리퀘스트에 대응하여, 등록된 상기 오퍼랜드 주소 정보와 상기 오퍼코드를 이용하여 상기 PIM 트랜잭션을 수행함을 특징으로 한

다.

[0014] 그리고, 본 발명의 PIM 컴퓨팅 시스템의 PIM 연산 오프로딩 방법은, CPU가 PIM 연산의 오프로딩을 위하여 메모리 리퀘스트를 전송하여서 DMA 엔진에 현재(Current) 디스크립터 주소와 테일(Tail) 디스크립터 주소를 저장하는 단계; 상기 PIM 연산 전, 상기 CPU가 상기 PIM 연산을 위한 모든 PIM 트랜잭션들을 DRAM의 메모리 어레이에 디스크립터들로서 저장하고 상기 DRAM의 PIM 디바이스에 디스크립터 주소 정보들을 등록하는 단계; 상기 DMA 엔진이 상기 PIM 연산을 위하여 상기 현재 디스크립터 주소부터 마지막의 상기 테일 디스크립터 주소까지 순서대로 디스크립터를 선택하고, 선택된 디스크립터에 대한 디스크립터 리퀘스트를 제공하여 디스크립터 정보를 리드하는 단계; 상기 DMA 엔진이 리드된 상기 디스크립터 정보에서 상기 오퍼랜드 주소 정보와 상기 오프 코드를 상기 PIM 디바이스에 제공하는 단계; 상기 PIM 디바이스가 상기 디스크립터 리퀘스트를 식별하고 상기 DMA 엔진에서 제공되는 상기 오퍼랜드 주소 정보와 상기 오프코드를 등록하는 단계; 상기 DMA 엔진이 리드된 상기 디스크립터 정보를 바탕으로 상기 PIM 트랜잭션을 발생시키는 상기 PIM 리퀘스트를 제공하는 단계; 상기 PIM 디바이스가 상기 PIM 리퀘스트를 식별하고 등록된 상기 오퍼랜드 주소 정보와 상기 오프코드를 이용하여 상기 PIM 트랜잭션을 수행하는 단계; 및 상기 현재 디스크립터 주소부터 마지막의 상기 테일 디스크립터 주소까지 상기 디스크립터들에 대하여 상기 디스크립터 리퀘스트와 상기 PIM 리퀘스트의 제공을 수행하여 상기 PIM 연산을 완료하는 단계;를 포함함을 특징으로 한다.

발명의 효과

[0015] 본 발명은 DMA 엔진의 설계 변경이나 기존의 기능을 변경하지 않고 캐시 계층(Cache Hierarchy)에 관련된 오버헤드 이슈없이 PIM 연산을 위한 대용량의 메모리 리퀘스트를 처리할 수 있는 효과가 있다.

[0016] 그리고, 본 발명은 PIM 연산을 위하여 저장되는 디스크립터의 여유 비트 필드를 활용하여 오프코드와 같이 오프로딩에 필요한 정보를 저장하고 PIM 트랜잭션을 위한 구성에 상기한 정보를 사용함으로써 일반 DMA 동작에 영향을 미치지 않고 PIM 연산을 효과적으로 수행할 수 있는 효과가 있다.

도면의 간단한 설명

[0017] 도 1은 본 발명의 PIM 컴퓨팅 시스템의 바람직한 실시예를 나타내는 블록도.
 도 2는 CPU가 DMA 엔진을 컨트롤 하기 위한 레지스터의 데이터 구조를 예시한 도면.
 도 3은 DMA 엔진을 예시한 상세 블록도.
 도 4는 본 발명에 의한 디스크립터의 데이터 구조를 예시한 도면.
 도 5는 디스크립터들의 구조를 예시한 도면.
 도 6은 DRAM을 예시한 상세 블록도.
 도 7은 도 6의 컨트롤 레지스터의 데이터 구조를 예시한 도면.
 도 8은 도 1의 시스템을 이용한 PIM 연산 오프로딩 방법을 예시한 흐름도.

발명을 실시하기 위한 구체적인 내용

[0018] 이하, 첨부된 도면을 참조하여 본 발명의 바람직한 실시예를 상세하게 설명한다. 본 명세서 및 특허청구범위에 사용된 용어는 통상적이거나 사전적 의미로 한정되어 해석되지 아니하며, 본 발명의 기술적 사항에 부합하는 의미와 개념으로 해석되어야 한다.

[0019] 본 명세서에 기재된 실시예와 도면에 도시된 구성은 본 발명의 바람직한 실시예이며, 본 발명의 기술적 사상을 모두 대변하는 것이 아니므로, 본 출원 시점에서 이들을 대체할 수 있는 다양한 균등물과 변형예들이 있을 수 있다.

[0020] 본 발명은 DMA 엔진을 PIM 연산의 오프로딩 엔진으로 사용함으로써 PIM 연산을 효과적으로 수행할 수 있는 시스템과 방법을 개시한다.

[0021] 본 발명의 실시예는 복수의 트랜잭션을 위한 대규모 PIM 연산을 CPU에서 DMA 엔진으로 오프로딩하며, DMA 엔진이 CPU와 커플링(Coupling)없이 메모리 리퀘스트에 대응한 대규모 PIM 연산을 수행하도록 구성될 수 있다.

[0022] 도 1을 참조하면, 본 발명의 PIM 컴퓨팅 시스템의 실시예는 CPU(10), 캐시 계층(12), DMA 엔진(20), 메모리 컨

트롤러(30) 및 DRAM(40)을 포함할 수 있다.

- [0023] 본 발명의 실시예는 메모리 버스(MB)를 이용하여 캐시 계층(12)을 통한 CPU(10)의 데이터 입출력, 메모리 컨트롤러(30)를 통한 DRAM(40)의 데이터 입출력 및 DMA 엔진(20)에 대한 메모리 리퀘스트의 오프로딩 등을 수행할 수 있다.
- [0024] 캐시 계층(12)은 메모리 버스(MB)를 통하여 입출력되는 CPU(10)의 데이터를 임시 저장하기 위하여 구성된 것으로 이해될 수 있고, CPU(10)의 입출력을 위한 데이터 양은 캐시 계층(12)의 캐시 용량에 의해 제한될 수 있다.
- [0025] 본 발명의 실시예는 대규모 PIM 연산을 위한 메모리 리퀘스트가 캐시 계층(12)의 상기한 캐시 용량에 의해 제한되는 것을 해소하기 위하여 DMA 엔진(20)에 PIM 연산을 오프로딩하도록 구성된다.
- [0026] 이를 위하여, CPU(10)는 DMA 엔진(20)에 PIM 연산의 오프로딩을 위한 메모리 리퀘스트를 제공할 수 있다. 대규모 PIM 연산의 오프로딩을 위한 메모리 리퀘스트는 도 2 및 도 3을 참조하여 후술한다.
- [0027] 또한, PIM 연산 전에, CPU(10)는 DRAM(40)에 PIM 연산에 필요한 정보를 제공할 수 있다. 여기에서, PIM 연산 전에 CPU(10)에서 DRAM(40)에 전달되는 정보는 DRAM(40)의 후술하는 메모리 어레이(60)에 저장되는 디스크립터들 및 DRAM(40)의 후술하는 PIM 인터페이스 유닛(52)의 컨트롤 레지스터 세트(56)에 저장되는 디스크립터 주소 정보들을 포함할 수 있다.
- [0028] DMA 엔진(20)에는 미리 설정된 데이터 구조를 갖는 컨트롤 레지스터가 설정될 수 있으며, CPU(10)의 메모리 리퀘스트에 대응하는 데이터가 컨트롤 레지스터에 저장될 수 있다. 상기한 컨트롤 레지스터의 데이터 구조는 도 2와 같이 예시될 수 있다.
- [0029] 도 2를 참조하면, DMA 엔진(20)의 컨트롤 레지스터는 제어(Control, EM10), 상태(Status, EM12), 현재 디스크립터 주소(Current Descriptor Address, EM20), 테일 디스크립터 주소(Tail Descriptor Address, EM22), 소스 주소(Source Address, EM30), 목적지 주소(Destination Address, EM32) 및 트랜스퍼 사이즈(Transfer size, EM34) 등을 각각 미리 설정된 주소 공간에 저장되도록 구성될 수 있다.
- [0030] 상기한 DMA 엔진(20)의 컨트롤 레지스터는 CPU(10)의 메모리 리퀘스트를 통하여 전달된 정보들으로써 설정될 수 있다.
- [0031] CPU(10)는 메모리 리퀘스트를 통하여 한 개 또는 둘 이상의 트랜잭션을 DMA 엔진(20)에 요청할 수 있다. 일 예로, 둘 이상의 트랜잭션을 위한 메모리 리퀘스트가 대규모 PIM 연산을 위한 메모리 리퀘스트로 이해될 수 있다.
- [0032] DMA 엔진(20)은 상기한 CPU(10)의 트랜잭션을 처리하기 위하여 도 3과 같이 구성된 것으로 예시될 수 있다. 도 3에서 컨트롤 레지스터는 도 2와 다르게 노멀 레지스터(NR)와 디스크립션 레지스터(DR)로 구분하여 도시된 것으로 이해될 수 있다. 그러나, 노멀 레지스터(NR)와 디스크립션 레지스터(DR)는 하나의 컨트롤 레지스터에 포함된 것으로 이해될 수 있다.
- [0033] 도 3을 참조하면, DMA 엔진(20)은 한 개의 트랜잭션을 처리하기 위한 데이터 무버(22)와 둘 이상의 트랜잭션을 처리하기 위한 디스크립터 컨트롤러(24)를 포함할 수 있다.
- [0034] CPU(10)가 한 개의 DMA 트랜잭션을 위한 메모리 리퀘스트를 제공하는 경우, DMA 엔진(20)은 메모리 리퀘스트를 통하여 전달되는 트랜잭션을 위한 정보를 노멀 레지스터(NR)에 등록하고, 데이터 무버(22)는 노멀 레지스터(NR)에 등록된 정보를 이용하여 한 개의 트랜잭션을 제어할 수 있다.
- [0035] 한 개의 DMA 트랜잭션을 위한 메모리 리퀘스트가 제공되는 경우, DMA 엔진(20)은 메모리 리퀘스트에 포함된 제어(Control, EM10), 상태(Status, EM12), 소스 주소(Source Address, EM30), 목적지 주소(Destination Address, EM32) 및 트랜스퍼 사이즈(Transfer size, EM34)를 노멀 레지스터(NR)의 미리 설정된 주소 공간에 등록할 수 있다.
- [0036] CPU(10)가 대규모 PIM 연산을 위하여 둘 이상의 트랜잭션을 위한 메모리 리퀘스트를 제공하는 경우, DMA 엔진(20)은 메모리 리퀘스트를 통하여 전달되는 트랜잭션을 위한 정보를 디스크립션 레지스터(DR)에 등록하고, 디스크립터 컨트롤러(24)는 디스크립션 레지스터(DR)에 등록된 정보를 이용하여 둘 이상의 트랜잭션을 순차적으로 제어할 수 있다.
- [0037] 둘 이상의 트랜잭션을 위한 메모리 리퀘스트를 제공하는 경우, DMA 엔진(20)은 메모리 리퀘스트에 포함된 디스크립션 모드의 제어(Control, EM10), 디스크립션 모드의 상태(Status, EM12), 현재 디스크립터 주소(Current

Descriptor Address, EM20) 및 테일 디스크립터 주소(Tail Descriptor Address, EM22)를 디스크립션 레지스터(DR)의 미리 설정된 주소 공간에 등록할 수 있다.

- [0038] 대규모 PIM 연산을 위한 메모리 리퀘스트에 대응하여, 디스크립터 컨트롤러(24)는 첫번째의 현재 디스크립터 주소 및 마지막의 테일 디스크립터 주소를 이용하여 메모리 리퀘스트에 해당하는 둘 이상의 트랜잭션을 순차적으로 제어할 수 있다.
- [0039] 즉, DMA 엔진(20)은 대규모 PIM 연산을 위한 메모리 리퀘스트에 대응하여 먼저 현재 디스크립터 주소에 해당하는 디스크립터 정보를 이용하여 처음의 PIM 연산을 수행하고, 현재 디스크립터 주소의 디스크립터에 등록된 다음 디스크립터 주소에 해당하는 디스크립터 정보를 이용하여 연속하는 PIM 연산을 수행하고, 상기한 방법으로 연속하는 PIM 연산을 수행한 후 마지막의 테일 디스크립터 주소에 해당하는 디스크립터 정보를 이용하여 마지막의 PIM 연산을 수행할 수 있다.
- [0040] 상기한 처음의 현재 디스크립터 주소, 연속하는 다음 디스크립터 주소 및 마지막의 테일 디스크립터 주소는 도 4 및 도 5를 참조하여 후술한다.
- [0041] 한편, 대규모의 메모리 리퀘스트에 대응한 복수의 PIM 트랜잭션을 제어하기 위하여, 본 발명의 실시예는 도 4 및 도 5와 같이 DRAM(40)에 PIM 연산을 위한 PIM 트랜잭션들을 디스크립터들로서 저장할 수 있다. 상기한 디스크립터들은 도 6을 참조하여 후술하는 DRAM(40)의 메모리 어레이(60)의 메모리 बैं크들에 분산 저장될 수 있다.
- [0042] DRAM(40)의 메모리 어레이(60)에 저장되는 각 디스크립터는 도 4와 같은 데이터 구조를 갖도록 구성될 수 있으며, 디스크립터들은 도 5와 같은 연결 구조를 가질 수 있다.
- [0043] 상기와 같이 DRAM(40)의 메모리 어레이(60)에 저장되는 디스크립터들은 상술한 바와 같이 PIM 연산 전에 CPU(10)에 의해 저장된 것으로 이해될 수 있다.
- [0044] 먼저, 각 디스크립터는 도 4와 같이 다음 디스크립터 주소(Next Descriptor Address)(DM10), 오퍼랜드 주소 정보, 트랜스퍼 사이즈(Transfer Size)(DM16) 및 트랜스퍼 상태(Transfer status)(DM18)를 포함하는 데이터 구조를 가질 수 있다. 오퍼랜드 정보는 소스 주소(Source Address)(DM12) 및 목적지 주소(Destination Address)(DM14)를 포함할 수 있다.
- [0045] 이 중, 트랜스퍼 상태(DM18)는 기타 활용을 위한 코드나 데이터를 등록하기 위한 여유 공간을 가질 수 있다.
- [0046] 각 디스크립터는 해당 트랜잭션을 위한 정보와 연결되는 순서의 트랜잭션을 위한 다음 디스크립터 주소(DM10)를 포함할 수 있다. 해당 트랜잭션을 위한 정보는 오퍼랜드 주소 정보, 트랜스퍼 사이즈(Transfer Size)(DM16) 및 트랜스퍼 상태(Transfer status)(DM18)가 해당되는 것으로 이해될 수 있다.
- [0047] 본 발명의 실시예는 트랜스퍼 상태(DM18)의 여유 공간에 오퍼코드(opcode, DM22)를 포함하도록 구성될 수 있으며, 트랜스퍼 완료(Transfer Completed, DM20) 등을 포함할 수 있다.
- [0048] 상기한 디스크립터들은 후속할 디스크립터에 대한 다음 디스크립터 주소(DM10)를 갖는다. 그러므로, 현재 디스크립터에 후속할 디스크립터는 현재 디스크립터에 포함된 다음 디스크립터 주소에 의해 지정될 수 있다.
- [0049] 도 5의 경우, 디스크립터 1이 첫번째의 현재 디스크립터 주소에 해당하는 것으로 이해될 수 있고, 디스크립터 2는 디스크립터 1의 다음 디스크립터 주소에 해당하는 것으로 이해될 수 있으며, 디스크립터 N이 마지막의 테일 디스크립터 주소에 해당하는 것으로 이해될 수 있다.
- [0050] 디스크립터들은 PIM 연산을 위하여 다음 디스크립터 주소에 의해 순차적으로 연결될 수 있다.
- [0051] 따라서, 대규모 메모리 리퀘스트가 DMA 엔진(20)에 오프로딩된 경우, DMA 엔진(20)은 메모리 리퀘스트를 통하여 제공된 첫번째의 현재 디스크립터 주소와 마지막의 테일 디스크립터 주소를 디스크립션 레지스터(DR)에 등록하고, 각 디스크립터의 다음 디스크립터 주소를 이용하여 디스크립터들의 트랜잭션들을 순차적으로 제어할 수 있다. 즉, DMA 엔진(20)은 도 5와 같이 첫번째의 현재 디스크립터 주소부터 마지막의 테일 디스크립터 주소까지 해당하는 디스크립터들에 대한 트랜잭션을 순서대로 제어할 수 있다.
- [0052] 그리고, DMA 엔진(20)은 트랜잭션 별로 디스크립터 리퀘스트와 PIM 리퀘스트를 순차적으로 제공할 수 있다.
- [0053] 이 중, 디스크립터 리퀘스트는 DMA 엔진(20)이 트랜잭션을 위한 정보를 DRAM(40)에서 리드하는 것으로 이해될 수 있다. 즉, DMA 엔진(20)은 메모리 리퀘스트에 대응하여 정해진 순서에 해당하는 디스크립터 주소 정보로써 DRAM(40)의 메모리 어레이(60)에 저장된 디스크립터 정보를 리드할 수 있다.

- [0054] 그리고, PIM 리퀘스트는 DMA 엔진(20)이 트랜잭션을 제어하는 것으로 이해될 수 있다. 즉, DMA 엔진(20)은 디스크립터 리퀘스트에 의해 리드된 디스크립터 정보로써 DRAM의 PIM 연산을 위한 트랜잭션을 제어할 수 있다.
- [0055] 상기한 DMA 엔진(20)의 제어에 의하여 트랜잭션을 수행하는 DRAM(40)의 상세한 구성은 도 6을 참조하여 설명한다.
- [0056] DRAM(40)은 메모리 버스(MB)에 연결된 메모리 컨트롤러(30)를 통하여 CPU(10) 및 DMA 엔진(20)과 통신하도록 구성되며, 명령 정보 및 주소 정보를 수신할 수 있고 데이터를 송수신할 수 있다.
- [0057] 상기한 메모리 컨트롤러(30)는 DRAM(40)에 대한 메모리 리퀘스트 및 데이터 입출력을 스케줄링하는 것으로 이해될 수 있다.
- [0058] 그리고, DRAM(40)은 명령 래치(42), 주소 래치(44) 및 데이터 입출력 회로(46)를 포함할 수 있다.
- [0059] 명령 래치(42)는 메모리 컨트롤러(30)를 통하여 제공되는 명령 정보 CMD를 수신 및 임시 저장한 후 출력하며, PIM 연산에 관한 명령 정보 CMD를 후술하는 PIM 디바이스(50)에 제공하도록 구성될 수 있다. PIM 연산을 위한 명령 정보 CMD는 DMA 엔진(20)에서 제공될 수 있으며, 메모리 어레이(60)를 제어하기 위한 बैं크 액티브(Bank Active), 리드(Read), 라이트(Write) 및 리프레시(Refresh) 등을 포함할 수 있다.
- [0060] 주소 래치(44)는 메모리 컨트롤러(30)를 통하여 제공되는 주소 정보 ADD를 수신 및 임시 저장한 후 출력하며, 주소 정보 ADD는 बैं크(Bank) 주소, 로오(Raw) 주소 및 컬럼(Column) 주소로 구성될 수 있다. PIM 연산을 위한 주소 정보 ADD는 DMA 엔진(20)에서 제공될 수 있으며, 예시적으로 디스크립터의 주소 정보 및 오퍼랜드의 주소 정보 등을 표현할 수 있다.
- [0061] 데이터 입출력 회로(Data I/O Circuit, 46)는 디스크립터들을 저장하기 위한 데이터 및 디스크립터 리퀘스트에 대응하는 데이터 등의 데이터 DATA의 입출력을 위하여 구성될 수 있다.
- [0062] 한편, DRAM(40)은 PIM 디바이스(50)와 메모리 어레이(60)를 더 포함할 수 있다.
- [0063] 메모리 어레이(60)는 복수의 메모리 बैं크들을 갖도록 구성되며, 메모리 बैं크들 각각은 정해진 저장 용량의 메모리 셀 어레이들을 포함하는 것으로 이해될 수 있다. 메모리 어레이(60)는 명령 래치(42)에서 제공되는 명령 정보 CMD 및 주소 래치(44)에서 제공되는 주소 정보 ADD를 수신하고, 입출력 경로를 통하여 데이터를 리드 또는 라이트하도록 구성될 수 있다.
- [0064] 예시적으로, 메모리 어레이(60)는 PIM 연산을 위한 트랜잭션들을 복수의 메모리 बैं크들에 디스크립터들로서 저장할 수 있다.
- [0065] 메모리 어레이(60)는 बैं크 액티브(Bank Active), 리드(Read), 라이트(Write) 및 리프레시(Refresh)와 같은 명령 정보 CMD에 의해 주소 정보 ADD에 해당하는 데이터를 처리할 수 있다.
- [0066] 특히, 메모리 어레이(60)는 DMA 엔진(20)에서 디스크립터 정보의 리드를 요청하는 경우 해당 주소의 디스크립터 정보를 리드하여 출력할 수 있다.
- [0067] 또한, 메모리 어레이(60)는 PIM 연산을 위한 트랜잭션의 경우 지정된 주소 정보의 오퍼랜드를 리드하여 PIM 디바이스(50)에 제공하거나 PIM 디바이스(50)에서 제공되는 PIM 연산 결과를 지정된 주소 정보로써 라이트할 수 있다.
- [0068] 한편, PIM 디바이스(50)는 DMA 엔진(20)의 제어에 의하여 트랜잭션을 수행하도록 구성된다.
- [0069] 보다 구체적으로, PIM 디바이스(50)는 DMA 엔진(20)의 디스크립터 리퀘스트에 대응하여 DMA 엔진(20)에서 제공되는 오퍼랜드 주소 정보와 오프 코드를 후술하는 PIM 컨트롤 레지스터 세트(56)에 등록할 수 있다. DMA 엔진(20)은 메모리 리퀘스트에 대응하여 메모리 어레이(60)에서 디스크립터 정보를 리드하며, 리드된 디스크립터 정보에서 오퍼랜드 주소 정보와 오프코드를 PIM 디바이스(50)에 제공할 수 있다.
- [0070] 그리고, PIM 디바이스(50)는 DMA 엔진(20)의 PIM 리퀘스트에 대응하여, 등록된 오퍼랜드 주소 정보와 오프코드를 이용하여 트랜잭션을 수행하도록 구성될 수 있다.
- [0071] 이를 위하여, PIM 디바이스(50)는 PIM 인터페이스 유니트(52) 및 PIM 엔진(54)을 포함하도록 구성될 수 있다.
- [0072] PIM 인터페이스 유니트(52)는 PIM 컨트롤 레지스터 세트(56)를 포함하며, 명령 정보 CMD 및 주소 정보 ADD를 수신하고, 디스크립터 리퀘스트에 대응하여 리드되는 디스크립터 정보를 수신할 수 있고, 연산 제어 신호 AC를 제

공하도록 구성될 수 있다.

- [0073] PIM 컨트롤 레지스터 세트(56)는 도 7과 같이 오퍼랜드 레지스터(REG A), 구성(Configuration) 레지스터(REG B) 및 디스크립터 레지스터(REG C)를 포함할 수 있다.
- [0074] 오퍼랜드 레지스터(REG A)는 오퍼랜드 주소 정보의 등록을 위한 것이다. 본 발명의 실시예는 DMA 엔진(20)에 의해 오퍼랜드 레지스터(REG A)를 구성한다. 그러므로, 해당되는 트랜잭션을 위한 하나의 오퍼랜드 주소가 등록될 수 있다.
- [0075] 구성 레지스터(REG B)는 오퍼코드의 등록을 위한 것이다. 본 발명의 실시예는 오퍼랜드 레지스터(REG A)에 등록된 하나의 오퍼랜드 주소에 해당하는 오퍼랜드의 동작을 정의하는 하나의 오퍼코드가 등록될 수 있다.
- [0076] 디스크립터 레지스터(REG C)는 디스크립터 주소 정보들의 등록을 위한 것이다. 디스크립터 레지스터(REG C)는 디스크립터 리퀘스트를 식별하기 위한 것이며, PIM 연산 전에 CPU(10)에서 제공되는 디스크립터 주소 정보들을 등록하는 것이다. 이때, 디스크립터 주소 정보는 디스크립터의 시작 주소와 크기를 포함할 수 있다. 디스크립터 레지스터(REG C)에 등록된 디스크립터 주소 정보는 연속한 영역에 저장된 디스크립터들을 위한 것일 수 있다. 연속되지 않은 디스크립터의 경우, CPU(10)가 DMA 엔진(20)을 사용하기 전에 디스크립터 레지스터(REG C)에 등록할 필요가 있다.
- [0077] 상기한 PIM 인터페이스 유닛(52)는 디스크립터 리퀘스트를 위하여 DMA 엔진(20)에서 주소 정보 ADD를 제공하면, DMA 엔진(20)의 주소 정보 ADD를 디스크립터 레지스터(REG C)의 주소 정보들과 비교하여 디스크립터 리퀘스트를 식별한다. DMA 엔진(20)의 주소 정보 ADD가 디스크립터 레지스터(REG C)의 주소 정보들 중 하나에 해당하면, PIM 인터페이스 유닛(52)는 DMA 엔진(20)이 디스크립터 리퀘스트를 제공한 것으로 식별할 수 있다.
- [0078] 디스크립터 리퀘스트를 식별한 경우, PIM 인터페이스 유닛(52)는 메모리 어레이(60)에서 DMA 엔진(20)으로 리드되는 디스크립터 정보 중 DMA 엔진(20)에서 제공되는 오퍼랜드 주소 정보를 오퍼랜드 레지스터(REG A)에 등록하고 오퍼코드를 구성 레지스터(REG B)에 등록하도록 구성될 수 있다.
- [0079] 또한, PIM 인터페이스 유닛(52)는 PIM 리퀘스트를 위하여 DMA 엔진(20)에서 주소 정보 ADD를 제공하면, DMA 엔진(20)의 주소 정보 ADD를 오퍼랜드 레지스터(REG A)의 오퍼랜드 주소 정보와 비교하여 PIM 리퀘스트를 식별한다. DMA 엔진(20)의 주소 정보 ADD와 오퍼랜드 레지스터(REG A)의 오퍼랜드 주소 정보가 일치하면, PIM 인터페이스 유닛(52)는 DMA 엔진(20)이 PIM 리퀘스트를 제공한 것으로 식별할 수 있다.
- [0080] PIM 인터페이스 유닛(52)는 PIM 리퀘스트인 경우, 오퍼랜드 레지스터(REG A)에 등록된 오퍼랜드 주소 정보 및 구성 레지스터(REG B)에 등록된 오퍼코드를 이용하여 트랜잭션의 실행을 위한 연산 제어 신호 AC를 제공하도록 구성될 수 있다. 뱅크 액티브(Bank Active), 리드(Read), 라이트(Write) 및 리프레시(Refresh)와 같은 명령 정보 CMD가 연산 제어 신호 AC의 제공에 이용될 수 있다.
- [0081] 한편, PIM 엔진(54)은 연산 제어 신호 AC에 대응하여 트랜잭션을 위한 연산을 수행하고, PIM 리퀘스트에 대응한 연산 결과를 제공하도록 구성될 수 있다.
- [0082] 연산 제어 신호 AC는 오퍼랜드 주소 정보와 오프 코드를 포함할 수 있으며, PIM 엔진(54)은 오퍼랜드 주소 정보에 해당하는 오퍼랜드를 리드하거나 PIM 연산 결과를 오퍼랜드 주소 정보에 라이트할 수 있고, 리드나 라이트를 결정하는 연산 속성은 오프 코드에 의해 결정될 수 있다.
- [0083] 트랜잭션의 완료되면, PIM 디바이스(50)는 PIM 리퀘스트 완료를 DMA 엔진(20)에 보고할 수 있다. 이를 위하여 예시적으로, PIM 엔진(54)은 트랜잭션이 종료되면 PIM 인터페이스 유닛(52)에 종료를 알릴 수 있고, PIM 인터페이스 유닛(52)는 데이터 경로를 통하여 DMA 엔진(20)에 트랜잭션 종료를 알릴 수 있다. 상기한 트랜잭션의 종료를 알리는 방법은 하나의 예시일 뿐이며 제작자에 의해 다양하게 DMA 엔진(20)에 트랜잭션의 종료가 전달될 수 있다.
- [0084] 한편, PIM 연산을 위한 트랜잭션이 처리되는 경우, 메모리 어레이(60)는 PIM 엔진(54)과 데이터 입출력을 위하여 연결될 필요가 있다.
- [0085] 이를 위하여, 스위치(70)가 구성될 수 있으며, 스위치(70)는 매칭 신호 MAT의 상태에 따라 메모리 어레이(60)와 데이터 입출력 회로(46) 간에 데이터 경로를 제공하거나 메모리 어레이(60)와 PIM 엔진(54) 간에 데이터 경로를 제공할 수 있다.
- [0086] 매칭 신호 MAT는 PIM 인터페이스 유닛(52)에 의해 제공될 수 있으며, PIM 인터페이스 유닛(52)는 PIM 리퀘

스트를 식별한 경우 메모리 어레이(60)와 PIM 엔진(54) 간에 데이터 경로를 형성하기 위한 매칭 신호 MAT를 제공할 수 있다.

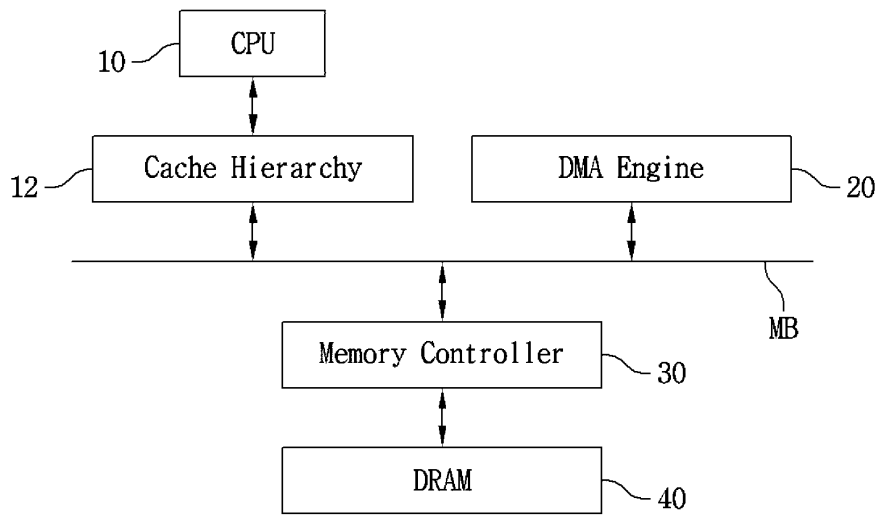
- [0087] 상술한 바와 같이 구성된 본 발명의 실시예에 의한 PIM 연산 방법은 도 8을 참조하여 설명될 수 있다.
- [0088] 먼저, PIM 연산을 위하여, CPU(10)는 PIM 연산에 필요한 트랜잭션들을 DRAM(40)의 메모리 어레이(60)에 디스크립터들로 저장한다(S2). PIM 연산 전에, 디스크립터들은 CPU(10)에서 제공되고 DRAM(40)의 메모리 어레이(60)에 저장될 수 있다.
- [0089] 또한, PIM 연산 전에, CPU(10)는 DRAM(40)의 메모리 어레이(60)에 저장될 디스크립터들의 디스크립터 주소 정보들을 PIM 인터페이스(52)에 등록할 수 있다(S2). 보다 구체적으로, 디스크립터 주소 정보들은 PIM 인터페이스(52)의 컨트롤 레지스터 세트(56)의 디스크립터 레지스터(REG C)에 등록될 수 있다.
- [0090] 그 후, CPU(10)는 DMA 엔진(20)에 메모리 리퀘스트를 전송할 수 있다(S4).
- [0091] 즉, CPU(10)는 대규모 PIM 연산을 위한 둘 이상의 트랜잭션들을 오프로딩하기 위한 메모리 리퀘스트를 DMA 엔진(20)에 제공하며, 메모리 리퀘스트를 통하여 PIM 연산을 위한 첫번째와 마지막 디스크립터들의 주소들을 알려줄 수 있다.
- [0092] 상기한 메모리 리퀘스트에 의하여 DMA 엔진(20)의 디스크립터 레지스터(DR)에 디스크립션 모드의 제어(EM10), 디스크립션 모드의 상태(EM12), 현재 디스크립터 주소(EM20) 및 테일 디스크립터 주소(EM22)가 등록될 수 있다. 현재 디스크립터 주소(EM20)가 첫번째 디스크립터 주소에 해당하고, 테일 디스크립터 주소(EM22)가 마지막 디스크립터 주소에 해당한다.
- [0093] 그 후, DMA 엔진(20)은 메모리 리퀘스트에 대응한 대규모 PIM 연산을 위한 트랜잭션들을 순차적으로 제어할 수 있다.
- [0094] 첫번째 트랜잭션은 현재 디스크립터 주소(EM20)에 해당하는 디스크립터에 대해 수행될 수 있다.
- [0095] 그러므로, DMA 엔진(20)은 현재 디스크립터 주소(EM20)에 해당하는 디스크립터에 대한 디스크립터 리퀘스트를 제공할 수 있으며, 디스크립터 리퀘스트에 대응하여 DRAM(40)의 메모리 어레이(60)에 저장된 현재 디스크립터 주소(EM20)에 해당하는 디스크립터 정보를 리드하고, 디스크립터 정보에 포함된 오퍼랜드 주소와 오퍼코드를 PIM 디바이스(50)의 PIM 인터페이스 유닛(52)에 제공할 수 있다(S6).
- [0096] 디스크립터 리퀘스트는 리드를 위한 명령 정보 CMD, 디스크립터에 해당하는 주소 정보 ADD를 포함하는 것으로 이해될 수 있으며, DRAM(40)의 메모리 어레이(60)는 주소 정보 ADD에 해당하는 디스크립터를 명령 정보 CMD에 대응하여 리드하여서 DMA 엔진(20)에 제공할 수 있다.
- [0097] 그리고, 스위치(70)는 매칭 신호 MAT의 상태에 따라 메모리 어레이(60)와 데이터 입출력 회로(46) 간에 데이터 경로를 제공하는 것으로 이해될 수 있다.
- [0098] 디스크립터 리퀘스트를 위하여 DMA 엔진(20)에서 주소 정보 ADD를 제공하면, PIM 디바이스(50)의 PIM 인터페이스 유닛(52)는 디스크립터 리퀘스트를 식별할 수 있고, DMA 엔진(20)이 제공하는 오퍼랜드 주소 정보와 오퍼코드를 오퍼랜드 레지스터(REG A)와 구성 레지스터(REG B)에 등록할 수 있다(S8).
- [0099] 보다 구체적으로, PIM 인터페이스 유닛(52)는 DMA 엔진(20)에서 제공되는 주소 정보 ADD를 디스크립터 레지스터(REG C)의 주소 정보들과 비교하여 디스크립터 리퀘스트를 식별할 수 있다. PIM 인터페이스 유닛(52)는 DMA 엔진(20)의 주소 정보 ADD가 디스크립터 레지스터(REG C)의 주소 정보들 중 하나에 해당하면 DMA 엔진(20)이 디스크립터 리퀘스트를 제공한 것으로 판단할 수 있다.
- [0100] 디스크립터 리퀘스트인 경우, DMA 엔진(20)은 메모리 어레이(60)에서 리드하는 디스크립터 정보 중 오퍼랜드 주소 정보와 오퍼코드를 제공하고, PIM 인터페이스 유닛(52)는 오퍼랜드 주소 정보를 오퍼랜드 레지스터(REG A)에 등록하고 오퍼코드를 구성 레지스터(REG B)에 등록할 수 있다.
- [0101] 그 후, DMA 엔진(20)은 리드한 디스크립터 정보를 바탕으로 PIM 리퀘스트를 제공할 수 있다(S10). PIM 리퀘스트는 디스크립터 리퀘스트에 의해 리드된 디스크립터 정보 중 오퍼랜드 주소 정보를 포함하는 것으로 이해될 수 있다.
- [0102] PIM 디바이스(50)는 상기한 PIM 리퀘스트를 식별하고 트랜잭션을 수행할 수 있다(S12).
- [0103] 이 중, PIM 리퀘스트 식별은 PIM 인터페이스 유닛(52)에 의해 수행되며, PIM 인터페이스 유닛(52)는 DMA 엔

진(20)에서 제공되는 주소 정보 ADD가 오퍼랜드 레지스터(REG A)에 등록된 오퍼랜드 주소 정보와 일치하면 DAM 엔진(20)이 PIM 리퀘스트를 제공한 것으로 식별할 수 있다.

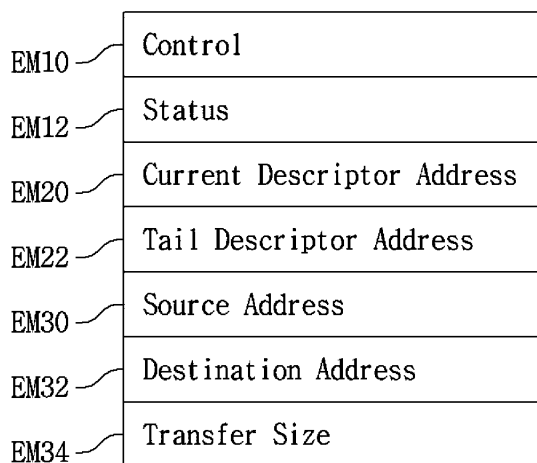
- [0104] PIM 리퀘스트가 식별되면, PIM 인터페이스 유닛(52)는 매칭 신호 MAT를 제공하고, 스위치(70)는 매칭 신호 MAT의 상태에 따라 메모리 어레이(60)와 PIM 엔진(54) 간에 데이터 경로를 제공하는 것으로 이해될 수 있다.
- [0105] 그리고, PIM 디바이스(50)에 의한 트랜잭션은 PIM 인터페이스 유닛(52)가 연산 제어 신호 AC를 제공함에 의해 수행될 수 있다. PIM 인터페이스 유닛(52)는 상술한 바와 같이 오퍼랜드 레지스터(REG A)에 등록된 오퍼랜드 주소 정보 및 구성 레지스터(REG B)에 등록된 오퍼코드를 이용하여 트랜잭션의 실행을 위한 상기한 연산 제어 신호 AC를 제공할 수 있다.
- [0106] 그리고, PIM 엔진(54)은 연산 제어 신호 AC에 대응하여 트랜잭션을 위한 연산을 수행하고 연산 결과를 제공할 수 있다. PIM 엔진(54)은 오퍼랜드 주소 정보에 해당하는 오퍼랜드를 리드하거나 PIM 연산 결과를 오퍼랜드 주소 정보에 라이트하는 등의 연산을 다양하게 수행될 수 있다. 그러므로, PIM 엔진(54)에서 수행되는 연산의 구체적인 예시는 생략한다.
- [0107] PIM 디바이스(50)의 트랜잭션이 완료되면, DMA 엔진(20)은 라이트(Write) 리퀘스트를 제공하여 트랜잭션이 완료된 메모리 어레이(60)의 디스크립터에 완료 비트(Complete Bit)를 라이트하고 PIM 인터페이스 유닛(52)의 컨트롤 레지스터 세트(56)를 클리어할 수 있다(S14).
- [0108] 상기와 같이 현재 디스크립터 주소(EM20)에 해당하는 첫번째 디스크립터의 트랜잭션이 완료되면, DMA 엔진(20)은 현재 디스크립터 주소(EM20)에 대한 디스크립터 리퀘스트를 통하여 리드한 디스크립터 정보에 포함된 다음 디스크립터 주소를 이용하여 다음 트랜잭션을 제어할 수 있다.
- [0109] 예시적으로, 상기한 과정에 의해 도 5의 디스크립터 1에 대한 트랜잭션이 완료되고, 디스크립터 1의 다음 디스크립터 주소가 디스크립터 2로 지정된 경우, DMA 엔진(20)은 디스크립터 2에 대한 트랜잭션을 제어할 수 있다.
- [0110] DMA 엔진(20)은 도 5의 디스크립터 2부터 마지막의 테일 디스크립터인 디스크립터 N까지 상기한 단계 S6 내지 S14를 반복할 수 있다(S16). 디스크립터 2부터 디스크립터 N까지의 디스크립터 리퀘스트와 PIM 리퀘스트는 상기한 단계 S6 내지 S14를 반복하여 수행되므로 이에 대한 중복 설명은 생략한다.
- [0111] 그 후, 마지막 테일 디스크립터의 트랜잭션이 완료(Complete)되면, DMA 엔진(20)은 CPU(10)에 PIM 연산의 완료에 따른 인터럽트(Interrupt) 신호를 제공할 수 있다(S18).
- [0112] 본 발명은 상술한 바와 같이 DMA 엔진의 설계 변경이나 기존의 기능을 변경하지 않고 오퍼코드를 갖도록 디스크립터들을 DRAM(40)의 메모리 어레이(60)에 저장하고 디스크립터들의 주소 정보를 DRAM(40)의 PIM 컨트롤 레지스터 세트(56)에 저장하는 것으로 PIM 연산을 위한 대용량의 메모리 리퀘스트를 처리할 수 있다.
- [0113] 따라서, 본 발명은 캐시 계층(Cache Hierarchy)에 관련된 오버헤드 이슈없이 대용량의 메모리 리퀘스트를 처리할 수 있는 효과가 있다.
- [0114] 또한, 본 발명은 PIM 연산을 위한 디스크립터의 여유 비트 필드를 활용하여 오퍼코드와 같이 오프로딩에 필요한 정보를 저장함으로써 일반 DMA 동작에 영향을 미치지 않고 PIM 연산을 효과적으로 수행할 수 있는 효과가 있다.

도면

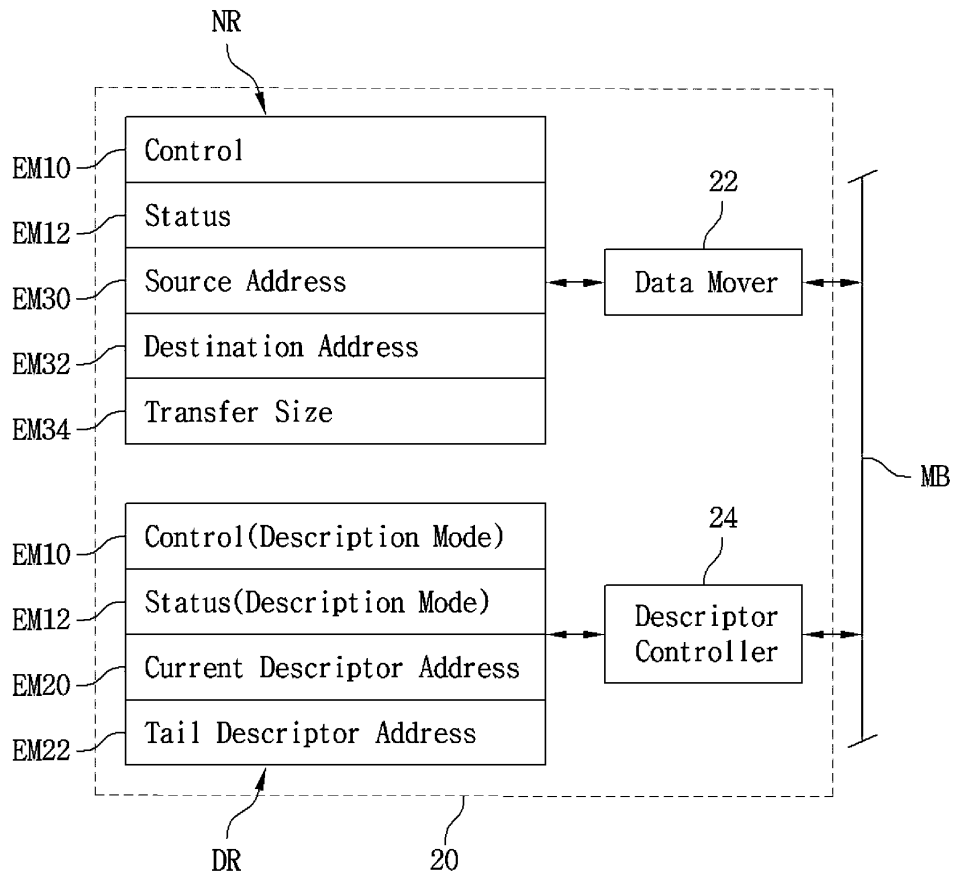
도면1



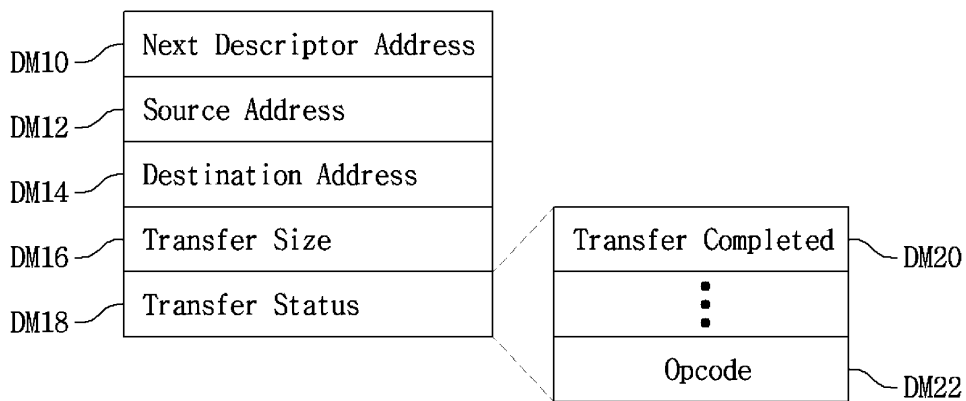
도면2



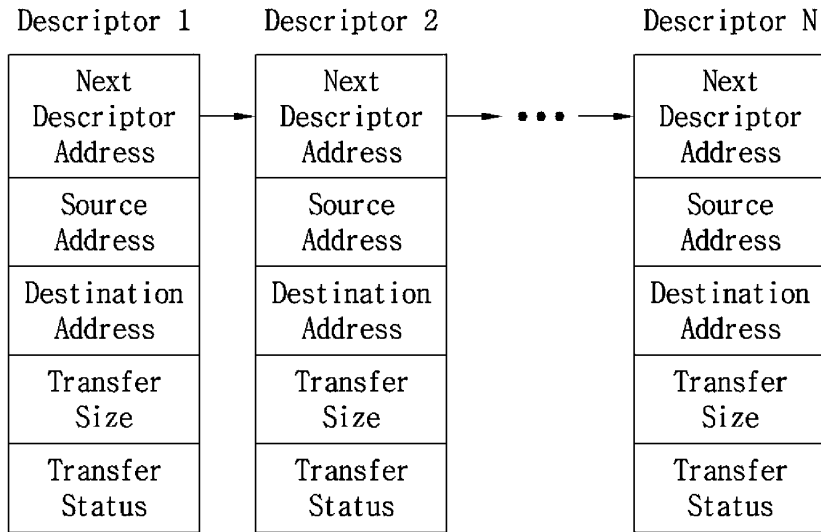
도면3



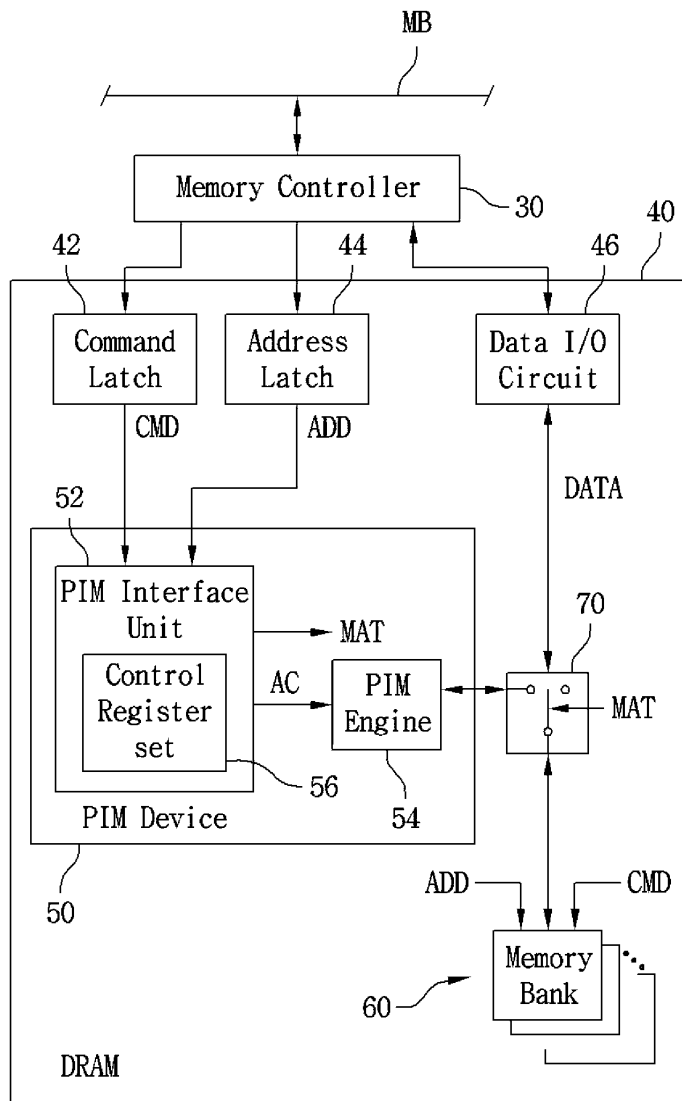
도면4



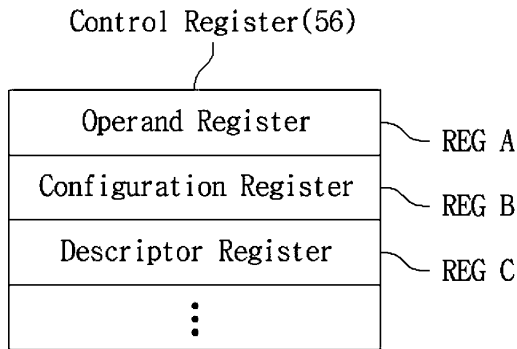
도면5



도면6



도면7



도면8

