(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2022년06월10일
(11) 등록번호 10-2408350
(24) 등록일자 2022년06월08일

(51) 국제특허분류(Int. Cl.)
G06T 1/20 (2018.01) *G06F 3/06* (2006.01)
G06T 1/60 (2006.01)
(52) CPC특허분류
G06T 1/20 (2013.01)
G06F 3/0631 (2013.01)
(21) 출원번호 10-2020-0080203
(22) 출원일자 2020년06월30일
심사청구일자 2020년06월30일
(65) 공개번호 10-2022-0001811
(43) 공개일자 2022년01월06일
(56) 선행기술조사문헌
KR1020160111122 A*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
연세대학교 산학협력단
서울특별시 서대문구 연세로 50 (신촌동, 연세대학교)
(72) 발명자
노원우
서울특별시 강남구 삼성로51길 35, 201동 1202호 (대치동, 래미안 대치 팰리스(2단지))
전원
서울특별시 서대문구 연희로14길 23, 301호(연희동)
(74) 대리인
민영준

전체 청구항 수 : 총 17 항 심사관 : 장재우

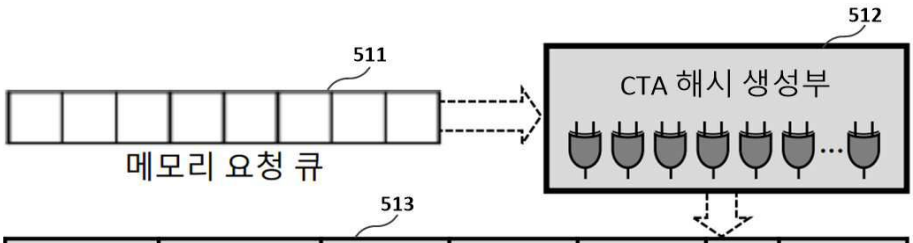
(54) 발명의 명칭 **에너지 효율을 개선할 수 있는 그래픽 처리 장치를 위한 메모리 컨트롤러 및 이의 메모리 제어 방법**

(57) 요 약

본 발명은 다수의 스트리밍 멀티프로세서와 다수의 메모리 컨트롤러를 포함하는 그래픽 처리 유닛과 각각 적어도 하나의 뱅크와 적어도 하나의 뱅크에 대응하는 기지정된 다수개의 로우 버퍼를 포함하는 다수의 메모리 칩이 포함된 다수의 메모리 채널로 구성된 메모리를 포함하는 그래픽 처리 장치의 메모리 컨트롤러에 있어서, 그래픽 처

(뒷면에 계속)

대 표 도 - 도6



Row Buffer ID	접근 Row 주소	CTA 0	CTA 1	CTA 2	…	CTA M-1
0	55A	0	153	20	-	0
1	Closed	3	2	35	-	235
…	-	-	-	-	-	-
N-1	600	215	0	165	-	0

리 장치에서 수행되어야 하는 프로그램을 구성하는 다수의 쓰레드 중 동일 연산을 수행하는 쓰레드가 그룹화된 다수의 와프를 각각 포함하는 다수의 CTA로부터 메모리 요청 명령을 인가받고, 메모리 요청 명령에 의해 지정되는 로우 주소와 로우 주소에 대응하는 뱅크에 대응하는 다수의 로우 버퍼의 식별자 중 하나를 맵핑하고, 다수의 CTA 각각이 맵핑된 로우 버퍼로의 접근 횟수를 카운트하여 카운트 값에 따라 지정된 개수의 로우 버퍼 각각을 다수의 CTA 중 적어도 하나의 CTA에 할당하여, 로우 버퍼에 대한 접근 및 충돌을 효율적으로 관리할 수 있으며, 에너지 소비를 저감시킬 수 있을 뿐만 아니라, 다수의 쓰레드 블록의 메모리 접근의 형평성을 보장할 수 있는 그래픽 처리 장치를 위한 메모리 컨트롤러 및 이의 메모리 제어 방법을 제공할 수 있다.

(52) CPC특허분류
G06F 3/0656 (2013.01)
G06F 3/0659 (2013.01)
G06T 1/60 (2013.01)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711110706
과제번호	2018R1A2A2A05018941
부처명	과학기술정보통신부
과제관리(전문)기관명	한국연구재단
연구사업명	후속연구지원(개인)
연구과제명	Warp 명령어 재사용 및 STT-MRAM을 활용한 GPU 데이터 공간 최적화 연구
기 여 율	1/1
과제수행기관명	연세대학교 산학협력단
연구기간	2020.03.01 ~ 2021.02.28

명 세 서

청구범위

청구항 1

다수의 스트리밍 멀티프로세서와 다수의 메모리 컨트롤러를 포함하는 그래픽 처리 유닛과 각각 적어도 하나의 뱅크와 상기 적어도 하나의 뱅크에 대응하는 기지정된 다수개의 로우 버퍼를 포함하는 다수의 메모리 칩이 포함된 다수의 메모리 채널로 구성된 메모리를 포함하는 그래픽 처리 장치의 메모리 컨트롤러에 있어서,

상기 그래픽 처리 장치에서 수행되어야 하는 프로그램을 구성하는 다수의 쓰레드 중 동일 연산을 수행하는 쓰레드가 그룹화된 다수의 와프를 각각 포함하는 다수의 CTA로부터 메모리 요청 명령을 인가받고, 상기 메모리 요청 명령에 의해 지정되는 로우 주소와 상기 로우 주소에 대응하는 뱅크에 대응하는 다수의 로우 버퍼의 식별자 중 하나를 맵핑하고, 상기 다수의 CTA 각각이 맵핑된 로우 버퍼로의 접근 횟수를 카운트하여 카운트 값에 따라 지정된 개수의 로우 버퍼 각각을 상기 다수의 CTA 중 적어도 하나의 CTA에 할당하며,

상기 메모리 컨트롤러는

상기 다수의 CTA로부터 메모리 요청 명령을 인가받아 저장하고, 저장된 메모리 요청 명령에 대응하는 로우 주소를 분석하여, 각 메모리 요청 명령이 지정하는 로우 주소와 기지정된 다수개의 로우 버퍼를 맵핑 및 로우 버퍼에 대한 접근 횟수를 카운트하여 다수의 CTA 각각이 기지정된 다수개의 로우 버퍼에 대한 접근 횟수를 카운트하여 저장하는 접근 패턴 분석부;

카운트된 다수의 CTA 각각이 기지정된 다수개의 로우 버퍼에 대한 접근 횟수에 기반하여, 상기 다수개의 로우 버퍼 각각을 상기 다수의 CTA 중 적어도 하나의 CTA에 할당하는 CTA 할당부; 및

상기 다수의 CTA의 메모리 요청 명령 중 이전 로우 버퍼가 맵핑되지 않은 로우에 대한 메모리 요청 명령에 대해 우선 순위를 부여하여, 상기 다수의 CTA의 메모리 요청 명령의 순서를 변경하는 RB 스케쥴러를 포함하는 그래픽 처리 장치를 위한 메모리 컨트롤러.

청구항 2

삭제

청구항 3

제1항에 있어서, 상기 접근 패턴 분석부는

상기 다수의 CTA의 메모리 요청 명령을 인가받아 저장하는 메모리 요청 큐;

상기 메모리 요청 큐에 저장된 다수의 메모리 요청 명령에 의해 지정된 로우 주소와 상기 기지정된 다수개의 로우 버퍼 및 상기 다수의 CTA 각각이 메모리 요청 명령으로 지정된 로우 주소에 접근하는 횟수가 저장되는 CTA-로우 버퍼 정보 테이블; 및

상기 메모리 요청 큐에 저장된 다수의 메모리 요청 명령에 의해 지정된 로우 주소와 상기 기지정된 다수개의 로우 버퍼를 맵핑하여 상기 CTA-로우 버퍼 정보 테이블을 생성하고, 상기 다수의 CTA 각각이 메모리 요청 명령으로 지정된 로우 주소에 접근하는 횟수를 카운트하여 생성된 상기 CTA-로우 버퍼 정보 테이블에 저장하는 CTA 해시 생성부를 포함하는 그래픽 처리 장치를 위한 메모리 컨트롤러.

청구항 4

제3항에 있어서, 상기 CTA 해시 생성부는

초기 상태에서는 다수의 CTA의 메모리 명령의 순서에 따라 접근해야 하는 접근 로우 주소를 기지정된 다수개 로우 버퍼의 식별자에 순차적으로 맵핑하여, 상기 CTA-로우 버퍼 정보 테이블을 생성하고, 이전 로우 버퍼의 식별자에 맵핑된 로우 주소에 대해 동일한 로우 주소로 접근하는 메모리 명령이 인가되면, 이전 맵핑된 로우 버퍼 식별자에 대응하는 CTA의 카운트값을 증가시키는 그래픽 처리 장치를 위한 메모리 컨트롤러.

청구항 5

제4항에 있어서, 상기 CTA 해시 생성부는

CTA의 메모리 명령에 의해 지정된 로우 주소가 맵핑된 로우 버퍼가 존재하지 않는 경우, 다수의 로우 버퍼의 식별자 중 하나의 로우 버퍼 식별자를 기지정된 방식으로 선택하여, 선택된 로우 버퍼에 맵핑된 로우 주소를 삭제하고, 새로이 입력된 메모리 명령에 따른 로우 주소를 로우 버퍼 식별자에 맵핑하여 상기 CTA-로우 버퍼 정보 테이블에 저장하는 그래픽 처리 장치를 위한 메모리 컨트롤러.

청구항 6

제5항에 있어서, 상기 CTA 해시 생성부는

로우 주소가 가장 먼저 맵핑된 로우 버퍼 식별자를 선택하여 선택된 로우 버퍼에 맵핑된 로우 주소를 삭제하는 그래픽 처리 장치를 위한 메모리 컨트롤러.

청구항 7

제4항에 있어서, 상기 CTA 할당부는

카운트된 카운트값 또는 카운트 비율에 따라 로우 버퍼를 하나의 CTA에 할당하거나 다수의 CTA에 공통으로 할당하는 그래픽 처리 장치를 위한 메모리 컨트롤러.

청구항 8

제4항에 있어서, 상기 기지정된 다수개의 로우 버퍼 각각은

할당된 CTA에 의해 맵핑된 로우의 데이터에 대해 칼럼 액세스와 활성화 또는 프리차지 중 적어도 하나가 수행되고, 할당되지 않은 CTA에 의해 맵핑된 로우의 데이터에 대해 칼럼 액세스가 수행되는 그래픽 처리 장치를 위한 메모리 컨트롤러.

청구항 9

제4항에 있어서, 상기 RB 스케쥴러는

상기 메모리 요청 큐에 저장된 메모리 요청 명령에서 현재 다수의 로우 버퍼가 활성화한 로우가 아닌 다른 로우에 대한 메모리 요청에 우선 순위를 주어 실행되어야 하는 명령의 순서를 변경하는 그래픽 처리 장치를 위한 메모리 컨트롤러.

청구항 10

다수의 스트리밍 멀티프로세서와 다수의 메모리 컨트롤러를 포함하는 그래픽 처리 유닛과 각각 적어도 하나의 뱅크와 상기 적어도 하나의 뱅크에 대응하는 기지정된 다수개의 로우 버퍼를 포함하는 다수의 메모리 칩이 포함된 다수의 메모리 채널로 구성된 메모리를 포함하는 그래픽 처리 장치의 메모리 제어 방법에 있어서,

상기 그래픽 처리 장치에서 수행되어야 하는 프로그램을 구성하는 다수의 쓰레드 중 동일 연산을 수행하는 쓰레드가 그룹화된 다수의 와프를 각각 포함하는 다수의 CTA로부터 메모리 요청 명령을 인가받는 단계;

상기 메모리 요청 명령에 의해 지정되는 로우 주소와 상기 로우 주소에 대응하는 뱅크에 대응하는 다수의 로우 버퍼의 식별자 중 하나를 맵핑하는 단계; 및

상기 다수의 CTA 각각이 맵핑된 로우 버퍼로의 접근 횟수를 카운트하여 카운트 값에 따라 지정된 개수의 로우 버퍼 각각을 상기 다수의 CTA 중 적어도 하나의 CTA에 할당하는 단계를 포함하고,

상기 맵핑하는 단계는

상기 다수의 CTA로부터 획득된 메모리 요청 명령을 저장하고, 저장된 메모리 요청 명령에 대응하는 로우 주소를 분석하여, 각 메모리 요청 명령이 지정하는 로우 주소와 기지정된 다수개의 로우 버퍼를 맵핑 및 로우 버퍼에 대한 접근 횟수를 카운트하여 다수의 CTA 각각이 기지정된 다수개의 로우 버퍼에 대한 접근 횟수를 카운트하는 단계;

카운트된 다수의 CTA 각각이 기지정된 다수개의 로우 버퍼에 대한 접근 횟수에 기반하여, 상기 다수개의 로우

버퍼 각각을 상기 다수의 CTA 중 적어도 하나의 CTA에 할당하는 단계; 및

상기 다수의 CTA의 메모리 요청 명령 중 이전 로우 버퍼가 맵핑되지 않은 로우에 대한 메모리 요청 명령에 대해 우선 순위를 부여하여, 상기 다수의 CTA의 메모리 요청 명령의 순서를 변경하는 단계를 포함하는 그래픽 처리 장치를 위한 메모리 제어 방법.

청구항 11

삭제

청구항 12

제10항에 있어서, 상기 카운트하는 단계는

상기 다수의 CTA의 메모리 요청 명령을 인가받아 메모리 요청 큐에 저장하는 단계;

상기 메모리 요청 큐에 저장된 다수의 메모리 요청 명령에 의해 지정된 로우 주소와 상기 기지정된 다수개의 로우 버퍼를 맵핑하여 CTA-로우 버퍼 정보 테이블을 생성하는 단계; 및

상기 다수의 CTA 각각이 메모리 요청 명령으로 지정된 로우 주소에 접근하는 횟수를 카운트하여 생성된 상기 CTA-로우 버퍼 정보 테이블에 저장하는 단계를 포함하는 그래픽 처리 장치를 위한 메모리 제어 방법.

청구항 13

제12항에 있어서, 상기 CTA-로우 버퍼 정보 테이블을 생성하는 단계는

초기 상태에서는 다수의 CTA의 메모리 명령의 순서에 따라 접근해야 하는 접근 로우 주소를 기지정된 다수개 로우 버퍼의 식별자에 순차적으로 맵핑하여, 상기 CTA-로우 버퍼 정보 테이블을 생성하는 그래픽 처리 장치를 위한 메모리 제어 방법.

청구항 14

제13항에 있어서, 상기 CTA-로우 버퍼 정보 테이블에 저장하는 단계는

이전 로우 버퍼의 식별자에 맵핑된 로우 주소에 대해 동일한 로우 주소로 접근하는 메모리 명령이 인가되면, 이전 맵핑된 로우 버퍼 식별자에 대응하는 CTA의 카운트값을 증가시키는 그래픽 처리 장치를 위한 메모리 제어 방법.

청구항 15

제14항에 있어서, 상기 카운트하는 단계는

CTA의 메모리 명령에 의해 지정된 로우 주소가 맵핑된 로우 버퍼가 존재하지 않는 경우, 다수의 로우 버퍼의 식별자 중 하나의 로우 버퍼 식별자를 기지정된 방식으로 선택하는 단계;

선택된 로우 버퍼에 맵핑된 로우 주소를 삭제하고, 새로이 입력된 메모리 명령에 따른 로우 주소를 로우 버퍼 식별자에 맵핑하여 상기 CTA-로우 버퍼 정보 테이블에 저장하는 단계를 포함하는 그래픽 처리 장치를 위한 메모리 제어 방법.

청구항 16

제15항에 있어서, 상기 선택하는 단계는

로우 주소가 가장 먼저 맵핑된 로우 버퍼 식별자를 선택하는 그래픽 처리 장치를 위한 메모리 제어 방법.

청구항 17

제14항에 있어서, 상기 적어도 하나의 CTA에 할당하는 단계는

카운트된 카운트값 또는 카운트 비율에 따라 로우 버퍼를 하나의 CTA에 할당하거나 다수의 CTA에 공통으로 할당하는 그래픽 처리 장치를 위한 메모리 제어 방법.

청구항 18

제14항에 있어서, 상기 기지정된 다수개의 로우 버퍼 각각은

할당된 CTA에 의해 맵핑된 로우의 데이터에 대해 칼럼 액세스와 활성화 또는 프리차지 중 적어도 하나가 수행되고, 할당되지 않은 CTA에 의해 맵핑된 로우의 데이터에 대해 칼럼 액세스가 수행되는 그래픽 처리 장치를 위한 메모리 제어 방법.

청구항 19

제14항에 있어서, 상기 순서를 변경하는 단계는

상기 메모리 요청 큐에 저장된 메모리 요청 명령에서 현재 다수의 로우 버퍼가 활성화한 로우가 아닌 다른 로우에 대한 메모리 요청에 우선 순위를 주어 실행되어야 하는 명령의 순서를 변경하는 그래픽 처리 장치를 위한 메모리 제어 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 그래픽 처리 장치를 위한 메모리 컨트롤러 및 이의 메모리 제어 방법에 관한 것으로, 에너지 효율을 개선할 수 있는 그래픽 처리 장치를 위한 메모리 컨트롤러 이의 메모리 제어 방법에 관한 것이다.

배 경 기 술

[0002] 일반적으로 그래픽 처리 장치의 그래픽 처리 유닛(Graphics Processing Unit: 이하 GPU)은 높은 병렬 연산 성능을 갖추기 위해 수천 개의 코어를 구비하고 있으며, 수십개의 코어가 하나의 스트리밍 멀티프로세서(Streaming Multiprocessor: 이하 SM)으로 그룹화되어 관리된다. 그리고 모든 SM에 공유되어 접근할 수 있는 라스트 레벨 캐시(Last-Level Cache)가 존재하며, 다수의 메모리 컨트롤러가 해당 라스트 레벨 캐시에 연결된다.

[0003] 각 메모리 컨트롤러는 대응하는 메모리 채널을 할당 받아, 메모리를 관리하고 데이터를 전달받는다. 각 메모리 채널은 다수의 메모리 칩(memory chip)으로 구성되며, 각 메모리 칩은 다수의 로우(row) 및 다수의 칼럼(column)을 포함하는 적어도 하나의 메모리 셀 어레이(memory cell array)로 구성되어 다수의 뱅크(bank)를 포함하는 계층 구조로 구성된다.

[0004] 이때, 각 메모리 칩에는 각 뱅크의 다수의 로우로 효율적으로 접근할 수 있도록 일종의 캐시 메모리 형태로 동작하는 로우 버퍼(row buffer)가 더 구비된다. 로우 버퍼를 구비하는 뱅크에서 원하는 데이터에 접근하기 위해서는, 목표하는 데이터가 저장된 셀 어레이에서 해당 로우 전체의 데이터를 로우 버퍼로 가져온 후, 이후 해당 로우에 대한 연속된 접근은 로우 히트(row hit)되어, 다시 로우의 데이터를 가져올 필요 없이 로우 버퍼가 곧바로 처리한다.

[0005] 그러나 해당 뱅크의 다른 로우의 데이터에 접근하기 위해서는 현재 로우 버퍼에 저장되어 있는 모든 데이터를 다시 셀 어레이로 복구하는 프리차지(precharge) 동작을 수행한 뒤, 접근하고자 하는 로우의 데이터를 다시 로우 버퍼로 가져와야 한다.

[0006] 결과적으로, 한번 로우 버퍼에 데이터를 가져온 뒤, 해당 로우에 연속된 접근을 많이 할수록, 로우 히트율이 증가되어 셀 어레이와 로우 버퍼 사이의 데이터 이동을 줄일 수 있어, 메모리 시스템의 에너지 효율을 향상시킬 수 있다. 다만 기존에는 메모리 시스템에서 로우 버퍼가 다수의 뱅크 각각에 대응하여 하나씩 구비됨에 따라 GPU의 메모리 컨트롤러는 단순히 로우 히트율을 최대로 높일 수 있도록 로우 버퍼를 관리하였다.

선행기술문헌

특허문헌

[0007] (특허문헌 0001) 한국 공개 특허 제10-2015-0035161호 (2015.04.06 공개)

발명의 내용

해결하려는 과제

[0008] 본 발명의 목적은 다수의 쓰레드 블록의 로우 접근 패턴에 따라 다수의 쓰레드 블록을 메모리에서 다수의 뱅크 각각에 대응하여 기지정된 개수로 구비된 다수의 로우 버퍼 각각에 할당하고, 할당된 다수의 쓰레드 블록만이 대응하는 로우 버퍼를 액티브 할 수 있도록 스케쥴링 함으로써, 로우 버퍼에 대한 접근 및 충돌을 효율적으로 관리할 수 있는 그래픽 처리 장치를 위한 메모리 컨트롤러 및 이의 메모리 제어 방법을 제공하는데 있다.

[0009] 본 발명의 다른 목적은 로우 버퍼에 대한 액티브 및 프리차지 동작 횟수를 줄여 에너지 소비를 저감시킬 수 있는 그래픽 처리 장치를 위한 메모리 컨트롤러 및 이의 메모리 제어 방법을 제공하는데 있다.

[0010] 본 발명의 또 다른 목적은 다수의 쓰레드 블록의 메모리 접근의 형평성을 보장할 수 있는 그래픽 처리 장치를 위한 메모리 컨트롤러 및 이의 메모리 제어 방법을 제공하는데 있다.

과제의 해결 수단

[0011] 상기 목적을 달성하기 위한 본 발명의 일 실시예에 따른 그래픽 처리 장치를 위한 메모리 컨트롤러는 다수의 스트리밍 멀티프로세서와 다수의 메모리 컨트롤러를 포함하는 그래픽 처리 유닛과 각각 적어도 하나의 뱅크와 상기 적어도 하나의 뱅크에 대응하는 기지정된 다수개의 로우 버퍼를 포함하는 다수의 메모리 칩이 포함된 다수의 메모리 채널로 구성된 메모리를 포함하는 그래픽 처리 장치의 메모리 컨트롤러에 있어서, 상기 그래픽 처리 장치에서 수행되어야 하는 프로그램을 구성하는 다수의 쓰레드 중 동일 연산을 수행하는 쓰레드가 그룹화된 다수의 와프를 각각 포함하는 다수의 CTA로부터 메모리 요청 명령을 인가받고, 상기 메모리 요청 명령에 의해 지정되는 로우 주소와 상기 로우 주소에 대응하는 뱅크에 대응하는 다수의 로우 버퍼의 식별자 중 하나를 맵핑하고, 상기 다수의 CTA 각각이 맵핑된 로우 버퍼로의 접근 횟수를 카운트하여 카운트 값에 따라 지정된 개수의 로우 버퍼 각각을 상기 다수의 CTA 중 적어도 하나의 CTA에 할당한다.

[0012] 상기 메모리 컨트롤러는 상기 다수의 CTA로부터 메모리 요청 명령을 인가받아 저장하고, 저장된 메모리 요청 명령에 대응하는 로우 주소를 분석하여, 각 메모리 요청 명령이 지정하는 로우 주소와 기지정된 다수개의 로우 버퍼를 맵핑 및 로우 버퍼에 대한 접근 횟수를 카운트하여 다수의 CTA 각각이 기지정된 다수개의 로우 버퍼에 대한 접근 횟수를 카운트하여 저장하는 접근 패턴 분석부; 카운트된 다수의 CTA 각각이 기지정된 다수개의 로우 버퍼에 대한 접근 횟수에 기반하여, 상기 다수개의 로우 버퍼 각각을 상기 다수의 CTA 중 적어도 하나의 CTA에 할당하는 CTA 할당부; 및 상기 다수의 CTA의 메모리 요청 명령 중 이전 로우 버퍼가 맵핑되지 않은 로우에 대한 메모리 요청 명령에 대해 우선 순위를 부여하여, 상기 다수의 CTA의 메모리 요청 명령의 순서를 변경하는 RB 스케쥴러를 포함할 수 있다.

[0013] 상기 접근 패턴 분석부는 상기 다수의 CTA의 메모리 요청 명령을 인가받아 저장하는 메모리 요청 큐; 상기 메모리 요청 큐에 저장된 다수의 메모리 요청 명령에 의해 지정된 로우 주소와 상기 기지정된 다수개의 로우 버퍼 및 상기 다수의 CTA 각각이 메모리 요청 명령으로 지정된 로우 주소에 접근하는 횟수가 저장되는 CTA-로우 버퍼 정보 테이블; 및 상기 메모리 요청 큐에 저장된 다수의 메모리 요청 명령에 의해 지정된 로우 주소와 상기 기지정된 다수개의 로우 버퍼를 맵핑하여 상기 CTA-로우 버퍼 정보 테이블을 생성하고, 상기 다수의 CTA 각각이 메모리 요청 명령으로 지정된 로우 주소에 접근하는 횟수를 카운트하여 생성된 상기 CTA-로우 버퍼 정보 테이블에 저장하는 CTA 해시 생성부를 포함할 수 있다.

[0014] 상기 CTA 해시 생성부는 초기 상태에서는 다수의 CTA의 메모리 명령의 순서에 따라 접근해야 하는 접근 로우 주소를 기지정된 다수개 로우 버퍼의 식별자에 순차적으로 맵핑하여, 상기 CTA-로우 버퍼 정보 테이블을 생성하고, 이전 로우 버퍼의 식별자에 맵핑된 로우 주소에 대해 동일한 로우 주소로 접근하는 메모리 명령이 인가되면, 이전 맵핑된 로우 버퍼 식별자에 대응하는 CTA의 카운트값을 증가시킬 수 있다.

[0015] 상기 CTA 해시 생성부는 CTA의 메모리 명령에 의해 지정된 로우 주소가 맵핑된 로우 버퍼가 존재하지 않는 경우, 다수의 로우 버퍼의 식별자 중 하나의 로우 버퍼 식별자를 기지정된 방식으로 선택하여, 선택된 로우 버퍼에 맵핑된 로우 주소를 삭제하고, 새로이 입력된 메모리 명령에 따른 로우 주소를 로우 버퍼 식별자에 맵핑하여 상기 CTA-로우 버퍼 정보 테이블에 저장할 수 있다.

[0016] 상기 CTA 해시 생성부는 로우 주소가 가장 먼저 맵핑된 로우 버퍼 식별자를 선택하여 선택된 로우 버퍼에 맵핑된 로우 주소를 삭제할 수 있다.

[0017] 상기 CTA 할당부는 카운트된 카운트값 또는 카운트 비율에 따라 로우 버퍼를 하나의 CTA에 할당하거나 다수의 CTA에 공통으로 할당할 수 있다.

[0018] 상기 기지정된 다수개의 로우 버퍼 각각은 할당된 CTA에 의해 맵핑된 로우의 데이터에 대해 칼럼 액세스와 활성

화 또는 프리차지 중 적어도 하나가 수행되고, 할당되지 않은 CTA에 의해 맵핑된 로우의 데이터에 대해 칼럼 액세스가 수행될 수 있다.

[0019] 상기 RB 스케쥴러는 상기 메모리 요청 큐에 저장된 메모리 요청 명령에서 현재 다수의 로우 버퍼가 활성화한 로우가 아닌 다른 로우에 대한 메모리 요청에 우선 순위를 주어 실행되어야 하는 명령의 순서를 변경할 수 있다.

[0020] 상기 다른 목적을 달성하기 위한 본 발명의 다른 실시예에 따른 그래픽 처리 장치를 위한 메모리 제어 방법은 다수의 스트리밍 멀티프로세서와 다수의 메모리 컨트롤러를 포함하는 그래픽 처리 유닛과 각각 적어도 하나의 뱅크와 상기 적어도 하나의 뱅크에 대응하는 기지정된 다수개의 로우 버퍼를 포함하는 다수의 메모리 칩이 포함된 다수의 메모리 채널로 구성된 메모리를 포함하는 그래픽 처리 장치의 메모리 제어 방법에 있어서, 상기 그래픽 처리 장치에서 수행되어야 하는 프로그램을 구성하는 다수의 쓰레드 중 동일 연산을 수행하는 쓰레드가 그룹화된 다수의 와프를 각각 포함하는 다수의 CTA로부터 메모리 요청 명령을 인가받는 단계; 상기 메모리 요청 명령에 의해 지정되는 로우 주소와 상기 로우 주소에 대응하는 뱅크에 대응하는 다수의 로우 버퍼의 식별자 중 하나를 맵핑하는 단계; 및 상기 다수의 CTA 각각이 맵핑된 로우 버퍼로의 접근 횟수를 카운트하여 카운트 값에 따라 지정된 개수의 로우 버퍼 각각을 상기 다수의 CTA 중 적어도 하나의 CTA에 할당하는 단계를 포함한다.

발명의 효과

[0021] 따라서, 본 발명의 실시예에 따른 그래픽 처리 장치를 위한 메모리 컨트롤러 및 이의 메모리 제어 방법은 다수의 쓰레드 블록의 로우 접근 패턴에 따라 다수의 쓰레드 블록을 메모리에서 다수의 뱅크 각각에 대응하여 기지정된 개수로 구비된 다수의 로우 버퍼 각각에 할당하고, 할당된 다수의 쓰레드 블록만이 대응하는 로우 버퍼를 액티브 할 수 있도록 스케쥴링 함으로써, 로우 버퍼에 대한 접근 및 충돌을 효율적으로 관리할 수 있다. 그러므로 로우 버퍼에 대한 액티브 및 프리차지 동작 횟수를 줄여 에너지 소비를 저감시킬 수 있을 뿐만 아니라, 다수의 쓰레드 블록의 메모리 접근의 형평성을 보장할 수 있다.

도면의 간단한 설명

[0022] 도 1은 본 발명의 일 실시예에 따른 메모리 컨트롤러를 포함하는 그래픽 처리 장치의 개략적 구조를 나타낸다.

도 2 및 도 3은 동일 쓰레드 그룹에 포함된 다수의 쓰레드와 서로 다른 쓰레드 그룹에 포함된 다수의 쓰레드의 로우 버퍼 접근 패턴을 시뮬레이션한 결과를 나타낸다.

도 4는 다수의 쓰레드 그룹 각각에 개별 로우 버퍼가 할당된 경우의 로우 로컬리티를 나타낸다.

도 5는 도 1의 메모리 컨트롤러의 구조를 나타낸다.

도 6은 도 5의 접근 패턴 분석부의 상세 구조의 일 예를 나타낸다.

도 7은 본 발명의 일 실시예에 따른 그래픽 처리 장치를 위한 메모리 제어 방법을 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0023] 본 발명과 본 발명의 동작상의 이점 및 본 발명의 실시에 의하여 달성되는 목적을 충분히 이해하기 위해서는 본 발명의 바람직한 실시예를 예시하는 첨부 도면 및 첨부 도면에 기재된 내용을 참조하여야만 한다.

[0024] 이하, 첨부한 도면을 참조하여 본 발명의 바람직한 실시예를 설명함으로써, 본 발명을 상세히 설명한다. 그러나, 본 발명은 여러 가지 상이한 형태로 구현될 수 있으며, 설명하는 실시예에 한정되는 것이 아니다. 그리고, 본 발명을 명확하게 설명하기 위하여 설명과 관계없는 부분은 생략되며, 도면의 동일한 참조부호는 동일한 부재임을 나타낸다.

[0025] 명세서 전체에서, 어떤 부분이 어떤 구성요소를 "포함"한다고 할 때, 이는 특별히 반대되는 기재가 없는 한 다른 구성요소를 제외하는 것이 아니라, 다른 구성요소를 더 포함할 수 있는 것을 의미한다. 또한, 명세서에 기재된 "...부", "...기", "모듈", "블록" 등의 용어는 적어도 하나의 기능이나 동작을 처리하는 단위를 의미하며, 이는 하드웨어나 소프트웨어 또는 하드웨어 및 소프트웨어의 결합으로 구현될 수 있다.

[0026] 도 1은 본 발명의 일 실시예에 따른 메모리 컨트롤러를 포함하는 그래픽 처리 장치의 개략적 구조를 나타내고, 도 2 및 도 3은 동일 쓰레드 그룹에 포함된 다수의 쓰레드와 서로 다른 쓰레드 그룹에 포함된 다수의 쓰레드의 로우 버퍼 접근 패턴을 시뮬레이션한 결과를 나타낸다.

[0027] 도 2는 도 1의 메모리 컨트롤러의 개략적 구조를 나타낸다.

[0028] 도 1을 참조하면, 본 실시예에 따른 그래픽 처리 장치(10)는 GPU(100)와 메모리(200)를 포함할 수 있다. GPU(100)는 높은 병렬 연산을 수행할 수 있도록 다수의 코어(Cores)를 구비하는 다수의 스트리밍 멀티프로세서(이하 SM)(111 ~ 11N)를 포함한다. 다수의 SM(111 ~ 11N) 각각은 단일 명령에 대해 다수의 쓰레드를 동시에 처리할 수 있도록 다수의 코어(core)(또는 스트림 프로세서(Stream processor)라고도 함)를 포함하고, 포함된 다수의 코어를 관리한다. 일 예로 각각의 SM(111 ~ 11N)은 수십개의 코어를 포함하도록 구성될 수 있으며, 다수의 쓰레드(Thread)를 쓰레드의 집합인 와프(Warp)(또는 웨이브프론트(wavefront)라고도 함) 단위로 병렬로 동시에 처리할 수 있다. 이때, SM(111 ~ 11N)은 와프를 단일 명령으로 처리하는 단일 명령 멀티 쓰레드(Single Instruction Multiple Threads: 이하 SIMT) 방식으로 실행될 수도 있다.

[0029] 그리고 다수의 SM(111 ~ 11N)은 와프의 컨텍스트 정보, 즉 명령을 저장하는 레지스터 파일(Resistor)과 다수의 코어가 쓰레드에 따른 연산을 수행하기 위한 데이터를 임시 저장하는 캐시 및 데이터의 호출 및 저장 명령을 실행하는 로드/스토어(Load/Store) 유닛을 더 포함할 수 있다.

[0030] 이러한 다수의 SM(111 ~ 11N)은 라스트 레벨 캐시(120)를 공유하여 이용하며, 다수의 메모리 컨트롤러(131 ~ 13N)는 메모리(200)의 다수의 메모리 채널(211 ~ 21N) 중 대응하는 메모리 채널을 할당받고, 할당된 메모리 채널을 관리한다. 즉 다수의 메모리 컨트롤러(131 ~ 13N) 각각은 할당된 메모리 채널에 저장된 데이터를 리드하여 라스트 레벨 캐시(120)로 전달하거나, 라스트 레벨 캐시(120)에 저장된 데이터를 메모리 채널로 전달하여 저장한다. 메모리 컨트롤러(131 ~ 13N)의 상세한 동작에 대해서는 후술하도록 한다.

[0031] 메모리(200)는 다수의 메모리 컨트롤러(131 ~ 13N)에 대응하는 다수의 메모리 채널(211 ~ 21N)을 포함하고, 다수의 메모리 채널(211 ~ 21N) 각각은 다수의 메모리 칩(301 ~ 30N)을 포함한다. 여기서는 일 예로 다수의 메모리 채널(211 ~ 21N) 각각이 N 개의 메모리 칩(301 ~ 30N)을 포함하는 것으로 도시하였으나, 메모리 칩(301 ~ 30N)의 개수는 다양하게 조절될 수 있다.

[0032] 여기서 다수의 메모리 칩(301 ~ 30N)은 일 예로 DRAM 메모리 칩으로 구현될 수 있으며, 각각 다수의 뱅크(BK)와 다수의 로우 버퍼(RB)를 포함한다. 다수의 뱅크 각각은 다수의 로우와 다수의 칼럼이 교차하는 위치 각각에 배치된 다수의 메모리 셀을 포함하는 적어도 하나의 메모리 셀 어레이로 구성되어 데이터를 저장한다.

[0033] 그리고 다수의 로우 버퍼(RB)는 다수의 메모리 컨트롤러(131 ~ 13N) 중 대응하는 메모리 컨트롤러에 의해 제어되어 다수의 뱅크(BK) 중 대응하는 뱅크(BK)에 저장된 데이터에 접근한다. 로우 버퍼(RB)는 메모리 컨트롤러에 의해 제어되어 다수의 메모리 셀에 저장된 데이터 중 요구되는 데이터가 저장된 로우를 활성화(Activate)하고, 활성화된 로우에 저장된 데이터를 일괄하여 인가받아 저장한다. 그리고 메모리 컨트롤러(131 ~ 13N)는 활성화되어 로우 버퍼(RB)에 저장된 로우의 데이터에 대해 칼럼 액세스(Column access)를 수행하여 특정 칼럼에 대응하는 데이터를 리드 또는 라이트한다. 여기서 칼럼 액세스 동작에 의해 소모되는 에너지를 칼럼 에너지라고 한다.

[0034] 한편, 메모리 컨트롤러(131 ~ 13N)는 활성화된 로우, 즉 이전 접근하여 로우 버퍼(RB)에 저장된 로우에 대한 접근이 더 이상 필요하지 않은 경우, 로우 버퍼(RB)에 저장된 데이터가 현재 활성화된 로우에 저장되도록 프리차지(Precharge) 동작을 수행한다. 여기서 로우 버퍼(RB)에 저장된 데이터는 활성화된 로우의 데이터가 획득된 이후, 칼럼 액세스 동작에 의해 특정 데이터가 다른 데이트로 라이트되어 변경된 데이터 일 수 있다.

[0035] 그리고 상기한 로우 버퍼가 특정 로우에 대해 수행하는 활성화 및 프리차지 동작 전체를 로우 액세스(Row access)라 하며, 로우 액세스 시에 소모되는 에너지를 로우 에너지라고 한다. 또한 로우 버퍼(RB)에 의해 하나의 로우가 활성화된 후, 프리차지되어 다시 재저장되기까지 해당 로우에 대해 칼럼 액세스가 수행된 횟수, 즉 해당 로우의 데이터가 리드 또는 라이트된 횟수를 로우 로컬리티(Row locality)라 하며, 로우 히트율(row hit rate)이 높을수록 로우 로컬리티 또한 증가하게 된다. 그리고 로우 로컬리티가 증가될수록 로우에 대한 활성화와 프리차지 동작을 수행하는 횟수를 줄일 수 있게 되므로, 로우 에너지 소모가 저감된다. 즉 메모리(200)의 에너지 소모가 저감될 수 있다.

[0036] 이에 메모리 컨트롤러(131 ~ 13N)는 GPU에서 수행되어야 할 어플리케이션 프로그램(400)을 분석하여, 프로그램(400)의 다수의 쓰레드를 그룹화함으로써 로우 로컬리티를 증가시킨다.

[0037] 일반적으로 GPU에서 수행될 프로그램(400)의 경우, 다수의 커널(Kernel)(410 ~ 4N0)로 구성되며, 다수의 커널(410 ~ 4N0) 각각은 다시 다수의 CTA(Cooperative Thread Array)(CTA0 ~ CTAN-1)로 구성된다. 여기서 다수의

CTA(CTA0 ~ CTAN-1) 각각은 다수의 쓰레드가 그룹화되어 구성된 쓰레드 블록이며, 다수의 CTA(CTA0 ~ CTAN-1) 각각의 다수의 쓰레드 중 동일한 연산을 수행하는 지정된 개수(32개 또는 64개와 같이 2의 승수에 대응하는 개수)의 쓰레드가 와프(warp)로 묶여 GPU(100)의 다수의 SM(111 ~ 11N) 중 특정 SM에서 실행된다.

[0038] 이때, 다수의 CTA(CTA0 ~ CTAN-1) 중 동일한 CTA에 포함된 다수의 쓰레드는 대부분 메모리에서 서로 인접한 영역에 저장된 데이터에 대한 연산을 수행하는 특성을 가지므로, 메모리 컨트롤러(131 ~ 13N)는 메모리(200)에서 서로 인접한 주소의 데이터에 액세스하는 경우가 빈번하게 발생하게 된다.

[0039] 즉 다수의 메모리 컨트롤러(131 ~ 13N) 각각은 특정 SM에서 실행되는 와프에 따라 다수의 메모리 칩(301 ~ 30N) 중 동일한 메모리 칩의 동일한 뱅크에 저장된 데이터에 빈번하게 액세스하게 된다.

[0040] 도 2 및 도 3은 GPGPU-Sim 시뮬레이터를 이용하여 다수의 CTA(CTA0 ~ CTAN-1) 중 동일 CTA의 다수의 쓰레드가 로우 버퍼(RB)에 접근하는 패턴과 서로 다른 CTA의 다수의 쓰레드가 로우 버퍼(RB)에 대해 접근하는 패턴을 시뮬레이션한 결과로서, 동일 CTA의 다수의 쓰레드의 접근 패턴을 인트라-CTA 로우 로컬리티(Intra-CTA row locality)라 하며, 서로 다른 CTA의 다수의 쓰레드의 접근 패턴을 인터-CTA 로우 로컬리티(Inter-CTA row locality)라 한다.

[0041] 도 2는 동일 CTA의 다수의 쓰레드의 접근 패턴인 인트라-CTA 로우 로컬리티에 대한 시뮬레이션 결과로서, 하나의 CTA에 3개의 곱셈(3 Multiplication) 연산에 대한 쓰레드가 포함된 경우(3M)와 행렬 곱셈(Matrix Multiplication) 연산에 대한 쓰레드가 포함된 경우(MM)의 시뮬레이션 결과를 함께 비교하여 도시하였다.

[0042] 도 2를 살펴보면, 다수의 CTA별(CTA ID)로 편차가 존재하지만, 동일 CTA의 쓰레드의 경우, 인접한 주소를 갖는 동일한 로우의 데이터에 접근하는 인트라-CTA 로우 로컬리티가 평균 6회 정도로 높게 나타남을 알 수 있다.

[0043] 도 3은 서로 다른 CTA의 다수의 쓰레드의 접근 패턴인 인트라-CTA 로우 로컬리티에 대한 시뮬레이션 결과로서, 도 3에서 다수의 막대 그래프 각각은 특정 프로그램이 실행되는 동안 하나의 로우가 활성화된 이후, 다시 프리차지되기 전까지 동일한 로우에 접근한 CTA의 개수를 나타낸다. 그리고 선 그래프는 활성화된 전체 로우 중 2개 이상의 CTA에 의해 공유 접근된 로우의 비율을 나타낸다. 시뮬레이션 결과 평균 2개의 CTA가 하나의 활성화된 로우에 접근하였으며, 평균 9.2%의 활성화된 로우가 다수의 CTA에 의해 공유 접근되었음이 확인되었다.

[0044] 이는 서로 다른 CTA가 활성화된 동일한 로우에 접근하는 인터-CTA 로우 로컬리티보다 동일 CTA가 활성화된 동일한 로우에 접근하는 인트라-CTA 로우 로컬리티가 상대적으로 매우 크게 나타남을 의미한다.

[0045] 따라서 다수의 CTA(CTA0 ~ CTAN-1) 각각에 대해 로우 버퍼(RB)를 개별적으로 할당할 수 있다면, 로우 로컬리티를 크게 향상시킬 수 있다.

[0046] 도 4에서는 도 2와 같이 동일 CTA의 다수의 쓰레드에 의한 로우 로컬리티를 나타내며, 다수의 CTA 각각에 대해 로우 버퍼가 개별적으로 할당된 경우의 로우 로컬리티와 다수의 CTA가 하나의 로우 버퍼를 공유하는 경우의 로우 로컬리티를 비교하여 나타내었다. 그리고 다수의 CTA가 하나의 로우 버퍼를 공유하는 경우, 메모리 컨트롤러(131 ~ 13N)는 FRFCFS(first-ready first-come first-serve) 정책에 따라 로우에 접근하는 경우를 가정하였다.

[0047] 도 4에 도시된 바와 같이, 다수의 CTA가 하나의 로우 버퍼(RB)를 공유하여 사용하는 경우에 비해, 다수의 CTA 각각에 개별적으로 로우 버퍼(RB)가 할당되면, 로우 로컬리티를 약 3배 정도 향상시킬 수 있음을 알 수 있다. 즉 로우 에너지를 대략 1/3으로 저감시킬 수 있다.

[0048] 그러나 다수의 메모리 칩(301 ~ 30N)의 다수의 뱅크(BK) 각각에 대해 CTA의 개수에 대응하는 수백 내지 수천개의 로우 버퍼를 구비하는 것은 현실적으로 불가능하다. 이에 본 발명에서는 각각의 뱅크(BK)에 대응하여 기지정된 다수개(예를 들면 5개)의 로우 버퍼(RB)가 구비된다. 다수의 뱅크(BK) 각각에 대해 다수개의 로우 버퍼(RB)가 구비됨에 따라 메모리 컨트롤러(131 ~ 13N)는 다수의 로우 버퍼(RB)가 할당될 CTA를 선별할 필요가 있다.

[0049] 특히 제한된 개수의 로우 버퍼(RB)를 이용하여 로우 로컬리티를 극대화할 수 있도록 CTA를 선별하기 위한 알고리즘이 필요하다.

[0050] 이에 본 실시예에서 메모리 컨트롤러(131 ~ 13N)는 다수의 CTA의 로우 접근 패턴을 분석하여, 분석된 로우 접근 패턴에 따라 다수의 CTA를 CTA 그룹으로 그룹화하고, CTA 그룹의 메모리 명령어의 우선 순위를 조절하여 로우 버퍼(RB)에 대한 로우 로컬리티를 극대화한다.

[0051] 여기서 다수의 메모리 컨트롤러(131 ~ 13N) 각각은 상기한 할당된 메모리 채널(211 ~ 21N)의 메모리 칩(301 ~ 30N)의 로우 버퍼(RB)에 대해서만 관리한다. 또한 다수의 뱅크(BK) 각각에 대해 기지정된 개수의 로우 버퍼(RB)가 구비되므로, 각 메모리 컨트롤러(131 ~ 13N)는 다수의 뱅크(BK) 각각에 대응하여 기지정된 개수의 로우 버퍼(RB)를 구분하여 제어한다. 이에 이하에서는 다수의 메모리 컨트롤러(131 ~ 13N) 중 하나가 대응하는 메모리 채널(211 ~ 21N)의 메모리 칩(301 ~ 30N)에서 특정 뱅크(BK)에 대응하여 구비된 다수의 로우 버퍼(RB)를 제어하는 방식에 대해 설명한다.

[0052] 도 5는 도 1의 메모리 컨트롤러의 구조를 나타내고, 도 6은 도 5의 접근 패턴 분석부의 상세 구조의 일 예를 나타낸다.

[0053] 도 5를 참조하면, 접근 패턴 분석부(510)와 CTA 할당부(520) 및 RB 스케쥴러(530)를 포함할 수 있다. 접근 패턴 분석부(510)는 도 6에 도시된 바와 같이, 메모리 요청 큐(511)와 CTA 해시 생성부(512) 및 CTA-로우 버퍼 정보 테이블(513)을 포함할 수 있다.

[0054] 메모리 요청 큐(511)는 다수의 CTA가 특정 뱅크(BK)에 저장된 데이터에 대해 요청하는 메모리 요청 명령을 인가되는 순서에 따라 저장하고, 선입선출법(first-input first-output: FIFO) 정책에 따라 먼저 인가된 명령을 우선 출력한다.

[0055] CTA 해시 생성부(512)는 메모리 요청 큐(511)에서 출력되는 메모리 요청 명령이 접근하고자 하는 뱅크(BK)의 데이터 주소를 기지정된 방식으로 분석하여 CTA-로우 버퍼 정보 테이블(513)에 다수의 CTA 각각의 로우 버퍼(RB)에 대한 접근 횟수를 카운트하여 기록한다.

[0056] CTA-로우 버퍼 정보 테이블(513)은 도 6에 도시된 바와 같이, 다수의 로우 버퍼(RB) 각각에 대한 식별자(Row Buffer ID)와 다수의 로우 중 활성화되어 데이터가 대응하는 로우 버퍼(RB)에 저장된 접근 로우 주소와 각 CTA가 로우 버퍼(RB)에 대응하는 접근 로우 주소에 접근한 횟수를 기록하여 저장한다.

[0057] 이때 CTA 해시 생성부(512)는 이전 로우에 대한 접근 내역이 존재하지 않는 초기 상태에서는 메모리 요청 큐(511)의 선입선출 정책에 대응하여 도착한 각 CTA의 메모리 명령의 순서에 따라 접근해야 하는 접근 로우 주소를 다수의 로우 버퍼(RB)의 식별자에 순차적으로 맵핑하여 기록한다. 또한 CTA 해시 생성부(512)는 이미 하나의 로우 버퍼(RB)의 식별자에 맵핑된 로우 주소에 대해 동일한 접근 로우 주소를 갖는 메모리 명령이 인가되면, 추가의 로우 버퍼 식별자를 맵핑하지 않고, 이전 맵핑된 로우 버퍼 식별자에 대응하는 CTA의 카운트값을 증가시킨다.

[0058] 그리고 모든 로우 버퍼(RB)에 접근 로우 주소가 맵핑되면, 이후 CTA 해시 생성부(512)는 각 CTA의 메모리 명령의 접근 로우 주소에 따라 CTA-로우 버퍼 정보 테이블(513)에 동일한 접근 로우 주소가 맵핑된 로우 버퍼 식별자에 대응하는 CTA의 카운트값을 증가시킨다.

[0059] 또한 CTA 해시 생성부(512)는 새로이 입력되는 CTA의 메모리 명령에 의해 지정된 로우 주소에 대해 CTA-로우 버퍼 정보 테이블(513)에서 현재 맵핑된 로우 버퍼가 존재하지 않는 경우, 다수의 로우 버퍼의 식별자 중 하나의 로우 버퍼 식별자를 선택하여 맵핑된 로우 주소를 삭제하고, 새로이 입력된 메모리 명령에 따른 로우 주소를 로우 버퍼 식별자에 맵핑하여 저장한다. 이때, CTA 해시 생성부(512)는 일 예로 다수의 로우 버퍼 식별자 중 현재 로우 주소가 가장 먼저 맵핑된 로우 버퍼 식별자를 선택할 수 있다.

[0060] 이는 선 도착 선 서비스(first-come first-serve: 이하 FIFS) 정책과 함께 라운드 로빈(round-robin) 정책을 기반으로 로우 버퍼 식별자와 로우 주소를 맵핑함으로써, 다수의 로우 버퍼에 최대한 많은 서로 다른 로우가 맵핑될 수 있도록 하기 위함이다.

[0061] 그러나 다른 예로서 CTA 해시 생성부(512)는 다수의 로우 버퍼 식별자 중 CTA에 의한 접근이 가장 오래된 로우 버퍼 식별자를 선택할 수도 있으며, 다수의 CTA에 의한 접근 횟수가 가장 작은 로우 버퍼 식별자를 선택할 수도 있다. 또한 CTA 해시 생성부(512)는 랜덤하게 로우 버퍼 식별자를 선택할 수도 있다.

[0062] CTA 해시 생성부(512)가 CTA-로우 버퍼 정보 테이블(513)에서 특정 로우 버퍼 식별자를 선택하여 맵핑된 로우 주소를 삭제하는 경우, 해당 로우 버퍼 식별자에 대응하여 저장된 각 CTA별 접근 카운트값 또한 함께 소거된다.

[0063] 한편 CTA 할당부(520)는 CTA-로우 버퍼 정보 테이블(513)을 참조하여, 기지정된 개수의 로우 버퍼(RB)를 CTA에 할당한다. CTA 할당부(520)는 CTA-로우 버퍼 정보 테이블(513)에서 각 로우 버퍼 식별자에 대해 맵핑된 CTA에 로우 버퍼(RB)를 순차적으로 할당한다.

[0064] CTA 할당부(520)에 의해 CTA에 할당된 로우 버퍼(RB)는 대응하는 CTA에 의해서만 매칭된 로우가 활성화 또는 프리차지되도록 하고, 나머지 CTA는 활성화되어 로우 버퍼(RB)에 저장된 데이터에 대해 칼럼 액세스만을 수행할 수 있도록 각 CTA의 로우 버퍼에 대한 권한을 설정한다.

[0065] 이후 CTA 할당부(520)는 CTA-로우 버퍼 정보 테이블(513)을 참조하여 다수의 로우 버퍼(RB) 각각에 대해 접근한 횟수, 즉 카운트값이 큰 CTA에 로우 버퍼(RB)를 할당한다. 이때, CTA 할당부(520)는 하나의 CTA에 하나의 로우 버퍼(RB)를 할당할 수 있을 뿐만 아니라, 다수의 CTA에 하나의 로우 버퍼(RB)를 공통으로 할당할 수도 있다.

[0066] 일 예로, 도 6에 도시된 CTA-로우 버퍼 정보 테이블(513)에 대해서, CTA 할당부(520)는 로우 버퍼 식별자에 따라 0번 로우 버퍼(RB0)를 가장 많이 접근한 CTA1에 할당하고, 1번 로우 버퍼(RB0)는 CTA M-1에 할당할 수 있다. 그리고 N-1번 로우 버퍼(RB N-1)은 CTA0과 CTA2에 공통으로 할당할 수 있다. 이는 CTA0과 CTA2가 모두 다른 CTA에 비해 N-1번 로우 버퍼(RB N-1)에 대한 접근 빈도가 높기 때문이다.

[0067] 여기서 CTA 할당부(520)가 다수의 로우 버퍼를 다수의 CTA 중 특정 CTA에 할당하기 위해 판단하는 카운트값의 기준은 다양하게 설정될 수 있다. 일 예로 CTA 할당부(520)는 기지정된 고정된 카운트값(예를 들면 100)을 기준으로 CTA에 대응하는 로우 버퍼(RB)를 할당할 수도 있으며, 기지정된 카운트 비율에 따라 로우 버퍼(RB)를 할당하도록 설정될 수도 있다.

[0068] 상기한 바와 같이 CTA 할당부(520)에 의해 로우 버퍼가 할당된 CTA는 할당된 로우 버퍼로 칼럼 액세스를 수행할 수 있을 뿐만 아니라, 로우 버퍼가 맵핑된 접근 로우 주소의 로우를 활성화 또는 프리차지하도록 할 수 있다. 그러나 다른 CTA는 칼럼 액세스만을 수행할 수 있다. 이는 인트라-CTA 로우 로컬리티를 향상시키면서도, 특정 CTA가 다수의 로우 버퍼를 독점하여 CTA간 형평성 문제가 발생할 수 있도록 하기 위함이다.

[0069] 그리고 CTA 할당부(520)는 CTA-로우 버퍼 정보 테이블(513)에서 특정 로우 버퍼 식별자에 맵핑된 로우 주소가 삭제되면, 선택된 로우 버퍼(RB)가 삭제된 로우 주소의 로우를 프리차지하도록 하여 이전 로우 버퍼(RB)에 저장된 데이터가 로우로 저장되도록 한다.

[0070] 이와 같이, CTA-로우 버퍼 정보 테이블(513)을 이용하여 각 로우에 대한 접근 횟수를 카운트하고, 카운트값에 따라 다수의 CTA에 기지정된 개수의 로우 버퍼(RB)를 할당하면, 로우 버퍼(RB)를 CTA의 개수만큼 구비하지 않더라도, 다수의 CTA가 제한된 개수의 로우 버퍼(RB)를 효과적으로 할당받아 이용함으로써 로우 로컬리티를 크게 향상시킬 수 있다. 즉 메모리(200)에서 소모되는 로우 에너지를 크게 저감시킬 수 있다.

[0071] 한편, RB 스케쥴러(530)는 메모리 요청 큐(511)에 저장된 메모리 요청 명령에서 현재 다수의 로우 버퍼(RB)가 활성화한 로우가 아닌 다른 로우에 대한 메모리 요청에 우선 순위를 주어 실행되어야 하는 명령의 순서에 대한 스케쥴링을 수행한다.

[0072] 기존의 메모리 컨트롤러(131 ~ 13N)에서는 각 뱅크(BK)에 대응하여 하나의 로우 버퍼(RB)만이 구비되었으므로, 로우 버퍼(RB)의 로우 히트율을 향상시키고자, 메모리 요청 큐(511)에 저장된 메모리 요청 명령 중 로우 버퍼(RB)에 의해 활성화된 로우에 대한 메모리 요청에 우선 순위를 주는 선 준비 선 도착 선 서비스(First-Ready First-Come First-Serve: 이하 FR-FCFS) 정책에 기반하여 메모리 요청 큐(511)에 저장된 명령의 순서를 변경하는 스케쥴링을 수행하였다.

[0073] 그러나 본 실시예에서는 각 뱅크(BK)에 대해 기지정된 개수의 다수의 로우 버퍼(RB)가 구비된다. 따라서 기존과 동일하게 FR-FCFS 정책에 기반하여 스케쥴을 수행하게 되면, 다수의 로우 버퍼(RB)가 존재함에도 하나의 로우 버퍼에 대한 명령만이 반복적으로 우선 처리되게 되어 명령어 실행 효율성이 낮아지게 된다. 이에 본 실시예에서는 로우 버퍼(RB)에 의해 활성화된 로우에 대한 메모리 요청이 아닌 다른 로우에 대한 메모리 요청에 우선 순위를 주어 메모리 요청 큐(511)에 저장된 명령의 순서를 변경하는 스케쥴링을 수행한다. 이 경우, 메모리 요청 큐(511)에 서로 다른 로우에 대한 접근 로우 주소가 배열되며, 이에 각 뱅크(BK)에 대응하여 구비되는 다수의 로우 버퍼(RB)를 최대한 활용할 수 있도록 되므로, 로우 버퍼의 활용성을 극대화할 수 있다.

[0074] 도 7은 본 발명의 일 실시예에 따른 그래픽 처리 장치를 위한 메모리 제어 방법을 나타낸다.

[0075] 도 7에 도시된 그래픽 처리 장치를 위한 메모리 제어 방법은 그래픽 처리 장치에 포함된 다수의 메모리 컨트롤러(131 ~ 13N) 각각에서 수행될 수 있다.

[0076] 도 1 내지 도 6을 참조하여, 도 7의 그래픽 처리 장치를 위한 메모리 제어 방법을 설명하면, 우선 그래픽 처리 장치(10)에서 실행되도록 인가된 어플리케이션 프로그램(400)을 구성하는 다수의 쓰레드 중 동일 연산을 수행하는 쓰레드가 그룹화된 다수의 와프를 포함하는 다수의 CTA를 인가받는다(S11). 이때 다수의 메모리 컨트롤러

(131 ~ 13N) 각각은 GPU(100)에 구비된 다수의 SM(111 ~ 11N) 중 대응하는 SM에서 실행되어야 하는 다수의 CTA를 인가받을 수 있다.

[0077] 다수의 CTA가 인가되면, 다수의 CTA가 인가된 순서에 따라 인가된 CTA에 포함된 쓰레드들의 메모리 요청 명령을 메모리 요청 큐(511)에 입력하여 저장한다(S12).

[0078] 그리고 메모리 요청 큐(511)에 저장된 메모리 요청 명령을 분석하여, CTA-로우 버퍼 정보 테이블(513)을 작성한다(S13). 여기서 CTA-로우 버퍼 정보 테이블(513)은 메모리 요청 큐(511)에 저장된 메모리 요청 명령에 의해 요청되는 데이터의 메모리 상의 로우 주소와 다수의 뱅크 각각에 대응하여 기지정된 다수개로 구비되는 로우 버퍼의 식별자가 맵핑되는 테이블이다.

[0079] 본 실시예에서 다수의 메모리 컨트롤러(131 ~ 13N) 각각은 메모리(200)의 다수의 메모리 채널(211 ~ 21N) 중 할당된 메모리 채널을 관리하며, 각 메모리 채널은 다수의 뱅크와 각 뱅크에 대해 기지정된 개수로 구비되는 다수의 로우 버퍼가 포함된 다수의 메모리 칩(301 ~ 30N)으로 구성된다. 그리고 다수의 뱅크 각각에는 다수의 로우와 다수의 칼럼이 교차하는 위치 각각에 데이터가 저장되는 메모리 셀을 포함한다.

[0080] 이에 메모리 요청 큐(511)에 저장된 메모리 요청 명령을 분석하여 CTA가 요청한 데이터의 뱅크별 로우 주소와 해당 뱅크에 대응하는 다수의 로우 버퍼에 대한 로우 버퍼 식별자를 맵핑하여 CTA-로우 버퍼 정보 테이블(513)을 작성할 수 있다.

[0081] CTA-로우 버퍼 정보 테이블(513)이 작성되면, 작성된 CTA-로우 버퍼 정보 테이블(513)에서 로우 주소에 맵핑된 로우 버퍼의 식별자와 메모리 요청 명령에 따라 다수의 로우 버퍼를 각각 메모리 요청 명령을 포함한 서로 다른 CTA에 할당한다(S14).

[0082] 여기서 로우 버퍼(RB)를 할당받은 CTA는 할당된 로우 버퍼(RB)가 맵핑된 로우에 대해 칼럼 액세스뿐만 아니라 활성화 또는 프리차지할 수 있으며, 할당되지 않은 CTA는 로우에 대해 칼럼 액세스만을 수행할 수 있게 된다.

[0083] 그리고 메모리 요청 큐(511)에 저장된 메모리 요청 명령에서 현재 다수의 로우 버퍼(RB)가 활성화한 로우가 아닌 다른 로우에 대한 메모리 요청에 우선 순위를 주어 메모리 요청 큐(511)에 저장된 명령의 순서를 변경하는 스케쥴링을 수행한다(S15).

[0084] 한편 스케쥴링된 메모리 요청 큐(511)에서 CTA의 명령어가 메모리에 저장된 데이터를 요청하는지 판별한다(S16). 만일 데이터를 요청하는 것으로 판별되면, 데이터가 저장된 로우에 대응하는 로우 버퍼(RB)가 할당된 CTA의 요청인지 판별한다(S17). 로우 버퍼(RB)가 할당된 CTA의 요청인 것으로 판별되면, 해당 CTA는 명령에 따라 해당 로우 버퍼(RB)에 대해 칼럼 액세스뿐만 아니라 로우에 대한 활성화와 프리차지 동작을 수행한다(S18). 그러나 로우 버퍼(RB)가 할당된 CTA가 아닌 다른 CTA의 요청인 경우, 칼럼 액세스에 해당하는 동작만을 수행한다(S19).

[0085] 그리고 각 CTA의 기지정된 개수의 로우 버퍼(RB) 각각에 대한 접근 횟수를 카운트한다(S20). 이후, 카운트된 각 CTA의 로우 버퍼(RB)에 대한 접근 횟수를 기반으로 카운트값이 기지정된 기준 카운트값 이상인 CTA 또는 카운트 비율이 기지정된 기준 카운트 비율 이상인 CTA에게 로우 버퍼(RB)를 재할당한다(S21).

[0086] 한편, 메모리 요청 큐(511)에 입력된 명령 중 현재 로우 버퍼(RB)가 매칭되어 활성한 로우를 제외한 다른 로우에 대한 요청이 존재하는지 판별한다(S22). 만일 다른 로우에 대한 요청이 존재한다면, 로우 버퍼(RB)가 요청된 로우가 활성화하도록 요청된 로우에 기반하여 CTA를 재할당한다(S23).

[0087] 그리고 그래픽 처리 장치에서 수행되어야 하는 프로그램이 종료되는지 판별한다(S24). 만일 프로그램이 종료되지 않은 것으로 판별되면, 다시 메모리 요청 큐(511)에 저장된 명령의 순서를 변경하는 스케쥴링을 수행한다(S15).

[0088] 본 발명에 따른 방법은 컴퓨터에서 실행시키기 위한 매체에 저장된 컴퓨터 프로그램으로 구현될 수 있다. 여기서 컴퓨터 판독가능 매체는 컴퓨터에 의해 액세스 될 수 있는 임의의 가용 매체일 수 있고, 또한 컴퓨터 저장 매체를 모두 포함할 수 있다. 컴퓨터 저장 매체는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 기타 데이터와 같은 정보의 저장을 위한 임의의 방법 또는 기술로 구현된 휘발성 및 비휘발성, 분리형 및 비분리형 매체를 모두 포함하며, ROM(판독 전용 메모리), RAM(랜덤 액세스 메모리), CD(컴팩트 디스크)-ROM, DVD(디지털 비디오 디스크)-ROM, 자기 테이프, 플로피 디스크, 광데이터 저장장치 등을 포함할 수 있다.

[0089] 본 발명은 도면에 도시된 실시예를 참고로 설명되었으나 이는 예시적인 것에 불과하며, 본 기술 분야의 통상의

지식을 가진 자라면 이로부터 다양한 변형 및 균등한 타 실시예가 가능하다는 점을 이해할 것이다.

[0090] 따라서, 본 발명의 진정한 기술적 보호 범위는 첨부된 청구범위의 기술적 사상에 의해 정해져야 할 것이다.

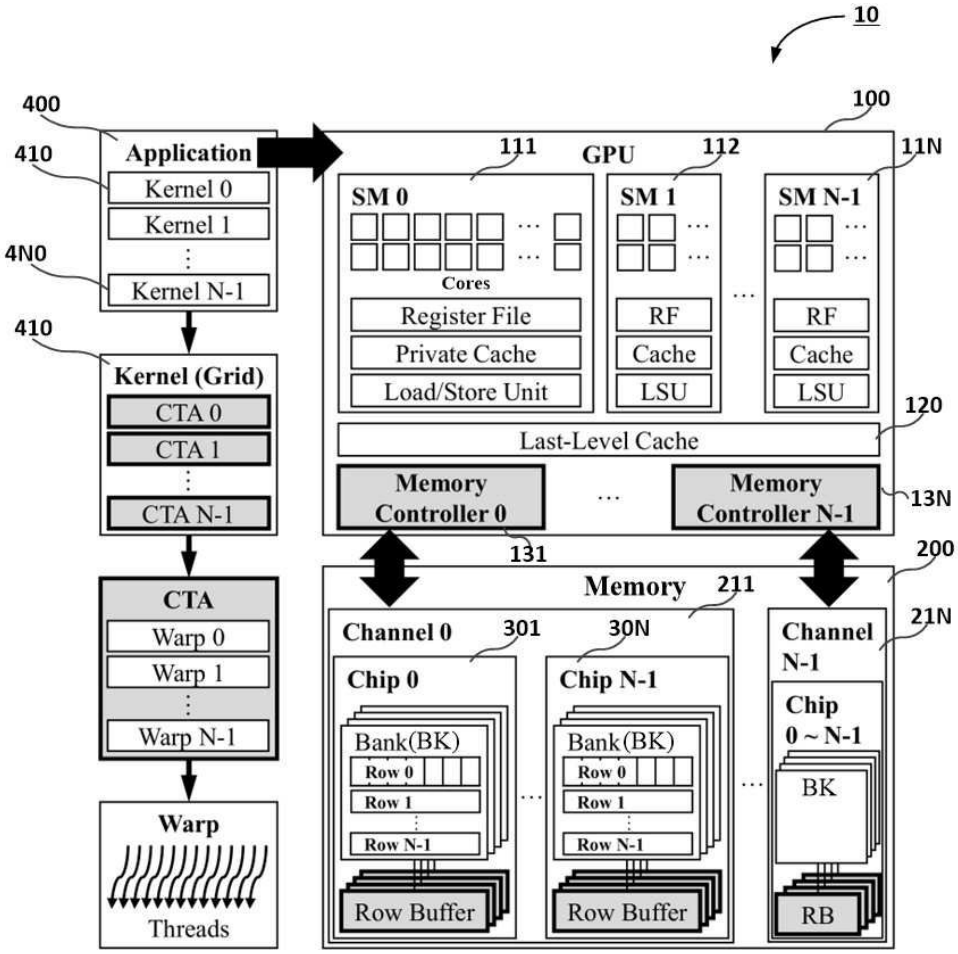
부호의 설명

[0091]

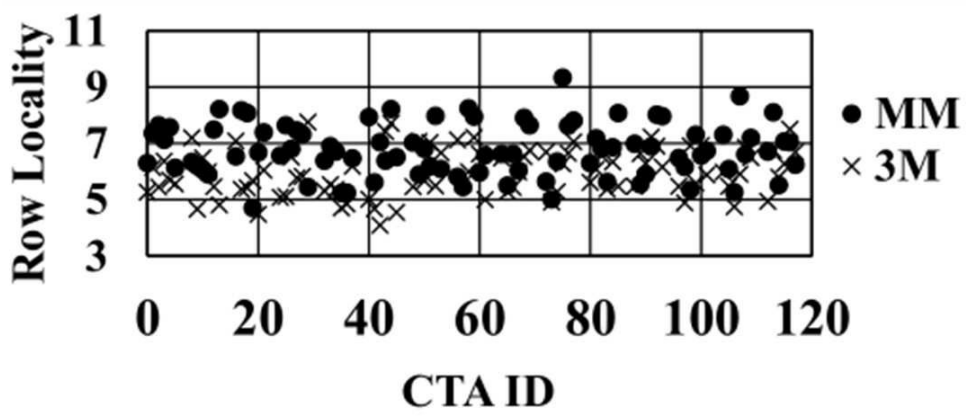
10: 그래픽 처리 장치	100: 그래픽 처리 유닛
111 ~ 11N: 스트리밍 멀티프로세서	120: 라스트 레벨 캐쉬
131 ~ 13N: 메모리 컨트롤러	200: 메모리
211 ~ 21N: 메모리 채널	301 ~ 30N: 메모리 칩
BK: 뱅크	RB: 로우 버퍼
400: 어플리케이션 프로그램	410 ~ 4N0: 커널

도면

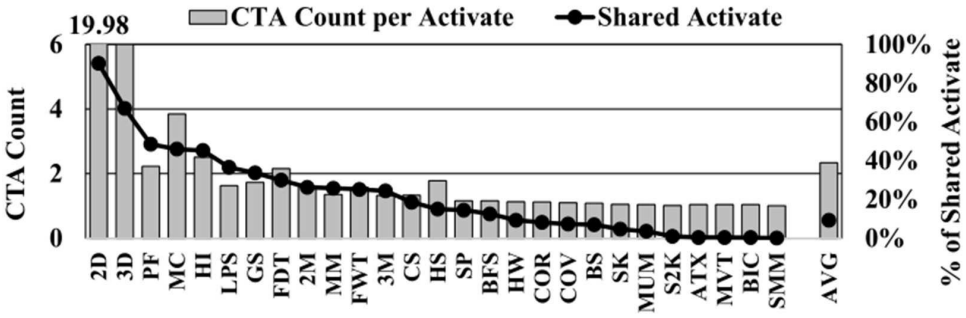
도면1



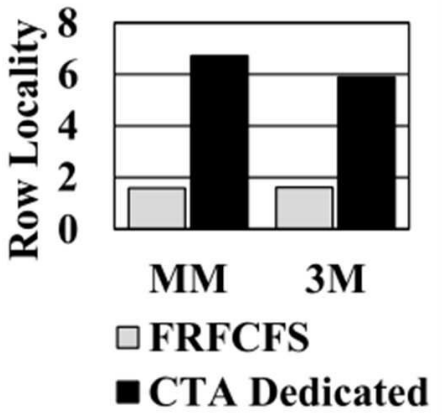
도면2



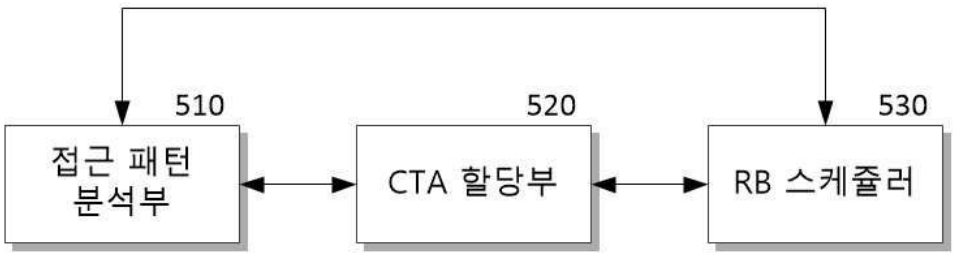
도면3



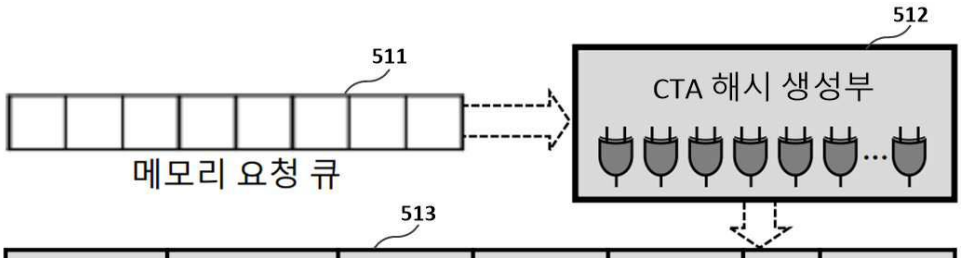
도면4



도면5



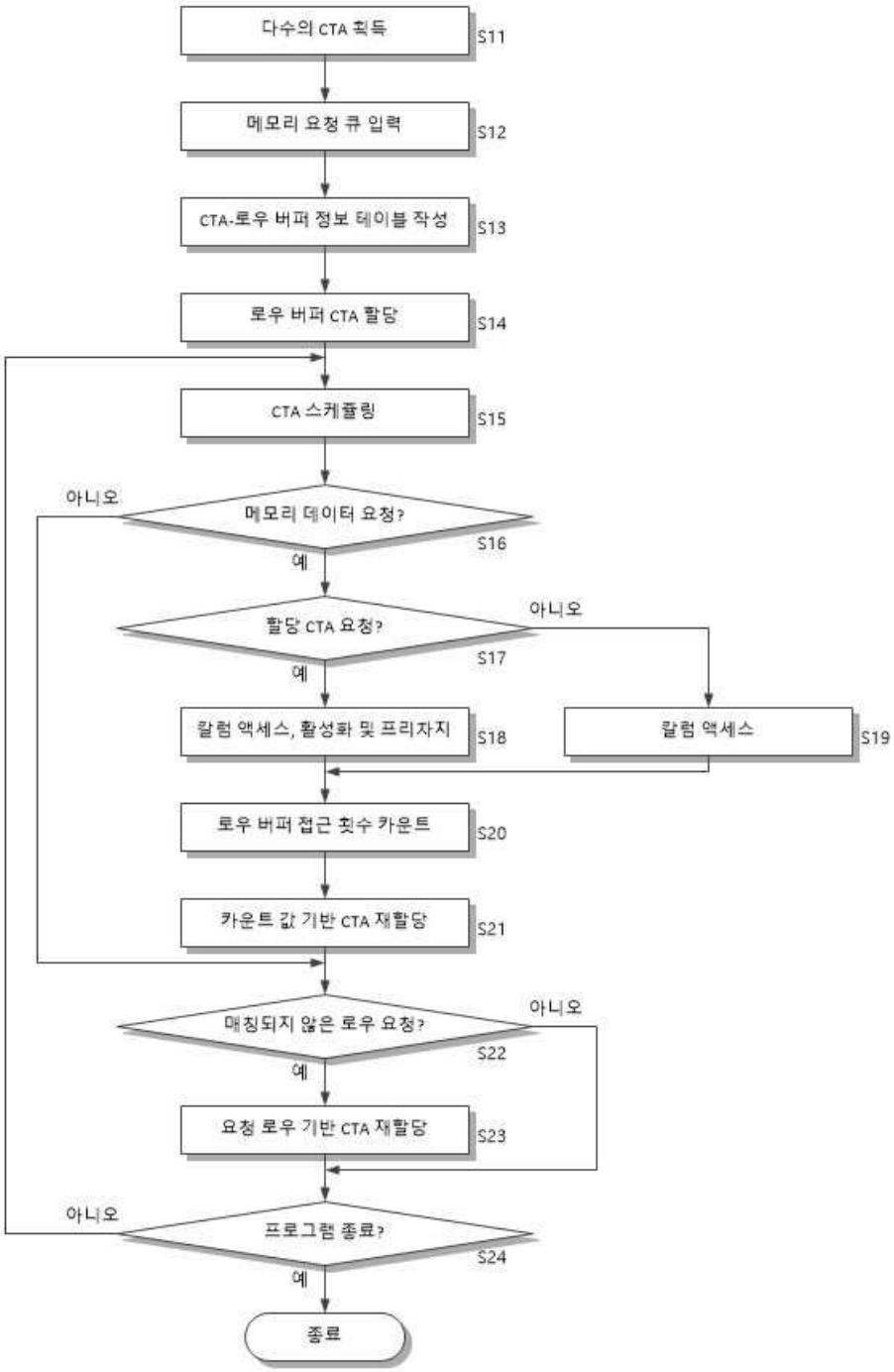
도면6



513

Row Buffer ID	접근 Row 주소	CTA 0	CTA 1	CTA 2	…	CTA M-1
0	55A	0	153	20	-	0
1	Closed	3	2	35	-	235
…	-	-	-	-	-	-
N-1	600	215	0	165	-	0

도면7


다수의 CTA 획득
S11
메모리 요청 큐 입력
S12
CTA-로우 버퍼 정보 테이블 작성
S13
로우 버퍼 CTA 할당
S14
CTA 스케쥴링
S15
아니오
메모리 데이터 요청?
S16
예
할당 CTA 요청?
아니오
S17
예
칼럼 액세스, 활성화 및 프리차지
S18
칼럼 액세스
S19
로우 버퍼 접근 횟수 카운트
S20
카운트 값 기반 CTA 재할당
S21
매칭되지 않은 로우 요청?
아니오
S22
예
요청 로우 기반 CTA 재할당
S23
아니오
프로그램 종료?
S24
예
종료