(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2020-0117301
(43) 공개일자 2020년10월14일

(51) 국제특허분류(Int. Cl.)
G06F 9/30 (2018.01) *G06F 9/38* (2006.01)
G06T 1/20 (2018.01)
(52) CPC특허분류
G06F 9/3012 (2013.01)
G06F 9/30007 (2013.01)
(21) 출원번호 10-2019-0039236
(22) 출원일자 2019년04월03일
심사청구일자 2019년04월03일

(71) 출원인
연세대학교 산학협력단
서울특별시 서대문구 연세로 50 (신촌동, 연세대학교)
(72) 발명자
노원우
서울특별시 강남구 삼성로51길 35, 201동 1202호 (대치동, 래미안 대치 팰리스(2단지))
박준현
서울특별시 송파구 올림픽로 212, C동 1002호 (잠실동, 갤러리아팰리스)
(74) 대리인
특허법인우인

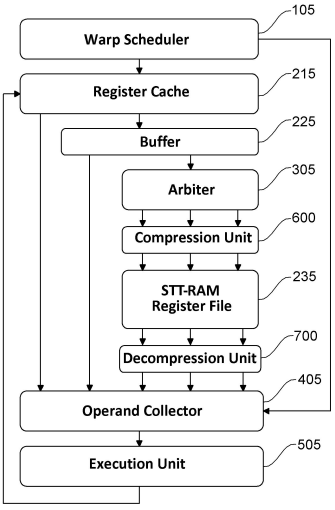
전체 청구항 수 : 총 14 항

(54) 발명의 명칭 **스핀 전달 토크 랜덤 액세스 메모리 기반의 계층적 레지스터 파일 장치**

(57) 요 약

본 실시예들은 범용 그래픽 처리 장치의 연산시 사용되는 레지스터 파일에 스핀 전달 토크 랜덤 액세스 메모리를 사용하여 에너지 효율을 높이고, 스핀 전달 토크 랜덤 액세스 메모리와 함께 레지스터 캐시와 버퍼를 계층적으로 사용함으로써, 누설 전류를 최소화하면서 쓰기 동작 전력을 줄이고 쓰기 지연을 해소할 수 있는 레지스터 파일 장치를 제공한다.

대 표 도 - 도3



(52) CPC특허분류
G06F 9/3885 (2013.01)
G06T 1/20 (2013.01)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711076330
부처명	산업통상자원부
과제관리(전문)기관명	한국산업기술평가관리원
연구사업명	실종아동등 신원확인을 위한 복합인지기술개발사업
연구과제명	현장출동 요원용 신원확인 정보처리 기술 개발
기 여 율	1/1
과제수행기관명	연세대학교
연구기간	2018.07.23 ~ 2019.04.22

명 세 서

청구범위

청구항 1

연산자와 피연산자를 포함하는 명령어의 집합인 데이터 처리 단위를 선택하는 스케줄러;

상기 피연산자에 대응하는 데이터를 계층적으로 저장하는 저장부;

상기 저장부로부터 상기 데이터를 수신하여 상기 피연산자의 준비 상태를 확인하는 피연산자 수집기;

상기 피연산자의 준비 상태에 따라 상기 데이터 처리 단위를 실행하는 연산부; 및

상기 연산부로부터 수신한 접근 요청을 저장하고 저장된 접근 요청을 상기 저장부에 할당하는 중계기

를 포함하는 레지스터 파일 장치.

청구항 2

제1항에 있어서,

상기 저장부는,

최근에 사용된 데이터를 저장하는 제1 저장부;

상기 제1 저장부로부터 방출된 데이터를 기 설정된 버퍼 주기 동안 저장하는 제2 저장부; 및

상기 제2 저장부로부터 수신한 데이터를 저장하는 제3 저장부를 포함하는 것을 특징으로 하는 레지스터 파일 장치.

청구항 3

제2항에 있어서,

상기 제3 저장부는 스핀 전달 토크 랜덤 액세스 메모리(Spin Transfer Torque-Random Access Memory, STT-RAM)로 구현된 것을 특징으로 하는 레지스터 파일 장치.

청구항 4

제2항에 있어서,

상기 제3 저장부에 저장되기 전에 기 설정된 압축 주기 동안 데이터를 압축하는 압축부 및 상기 압축된 데이터를 해제하는 해제부를 추가로 포함하는 것을 특징으로 하는 레지스터 파일 장치.

청구항 5

제4항에 있어서,

상기 압축부는 복수의 데이터에 대해서 기준 데이터를 설정하고 상기 기준 데이터와의 차이로 표현하는 방식으로 데이터의 비트 수를 감소시키고, 상기 해제부는 상기 기준 데이터에 상기 기준 데이터와의 차이를 더하여 복원하는 것을 특징으로 하는 레지스터 파일 장치.

청구항 6

제4항에 있어서,

상기 버퍼 주기를 세고 상기 압축 주기를 세는 카운터를 추가로 포함하는 것을 특징으로 하는 레지스터 파일 장치.

청구항 7

제4항에 있어서,

상기 접근 요청이 쓰기 요청이면, 상기 제2 저장부는 상기 제1 저장부로부터 방출된 데이터를 상기 버퍼 주기 및 상기 압축 주기 동안 저장하고, 상기 압축부는 상기 제2 저장부로부터 수신한 데이터를 압축한 후 상기 제3 저장부는 상기 압축된 데이터를 저장하는 것을 특징으로 하는 레지스터 파일 장치.

청구항 8

제4항에 있어서,

상기 접근 요청이 읽기 요청이고 상기 읽기 요청에 해당하는 피연산자가 상기 제2 저장부에 존재하지 않으면, 상기 제3 저장부를 확인하고,

상기 해제부는 상기 피연산자에 대응하는 압축된 데이터를 해제한 후 상기 제3 저장부는 상기 피연산자 수집기로 전송하는 것을 특징으로 하는 레지스터 파일 장치.

청구항 9

제2항에 있어서,

상기 접근 요청이 쓰기 요청이고 상기 쓰기 요청에 해당하는 피연산자가 상기 제1 저장부에 존재하면, 상기 제1 저장부는 새로운 데이터로 덮어 쓰는 것을 특징으로 하는 레지스터 파일 장치.

청구항 10

제9항에 있어서,

상기 접근 요청이 쓰기 요청이고 상기 쓰기 요청에 해당하는 피연산자가 상기 제1 저장부에 존재하지 않으면, 상기 제1 저장부는 기존 데이터를 상기 제2 저장부로 방출한 후 상기 제1 저장부는 새로운 데이터로 덮어 쓰는 것을 특징으로 하는 레지스터 파일 장치.

청구항 11

제2항에 있어서,

상기 접근 요청이 읽기 요청이고 상기 읽기 요청에 해당하는 피연산자가 상기 제1 저장부에 존재하면, 상기 제1 저장부는 상기 피연산자에 대응하는 데이터를 상기 피연산자 수집기로 전송하는 것을 특징으로 하는 레지스터 파일 장치.

청구항 12

제11항에 있어서,

상기 접근 요청이 읽기 요청이고 상기 읽기 요청에 해당하는 피연산자가 상기 제1 저장부에 존재하지 않으면, 상기 제2 저장부를 확인하고,

상기 읽기 요청에 해당하는 피연산자가 상기 제2 저장부에 존재하면, 상기 피연산자에 대응하는 데이터를 상기 피연산자 수집기로 전송하는 것을 특징으로 하는 레지스터 파일 장치.

청구항 13

그래픽 처리 장치에 있어서,

데이터를 병렬 처리하는 멀티 프로세서; 및

상기 데이터를 저장하는 레지스터 파일 장치를 포함하며,

상기 레지스터 파일 장치는,

연산자와 피연산자를 포함하는 명령어의 집합인 데이터 처리 단위를 선택하는 스케줄러;

상기 피연산자에 대응하는 데이터를 계층적으로 저장하는 저장부;

상기 저장부로부터 상기 데이터를 수신하여 상기 피연산자의 준비 상태를 확인하는 피연산자 수집기;

상기 피연산자의 준비 상태에 따라 상기 데이터 처리 단위를 실행하는 연산부; 및

상기 연산부로부터 수신한 접근 요청을 저장하고 저장된 접근 요청을 상기 저장부에 할당하는 중계기를 포함하는 것을 특징으로 하는 그래픽 처리 장치.

청구항 14

제13항에 있어서,

상기 저장부는,

최근에 사용된 데이터를 저장하는 제1 저장부;

상기 제1 저장부로부터 방출된 데이터를 기 설정된 버퍼 주기 동안 저장하는 제2 저장부; 및

상기 제2 저장부로부터 수신한 데이터를 저장하는 제3 저장부를 포함하는 것을 특징으로 하는 그래픽 처리 장치.

발명의 설명

기 술 분 야

[0001] 본 발명이 속하는 기술 분야는 레지스터 파일 및 그래픽 처리 장치에 관한 것이다.

배 경 기 술

[0002] 이 부분에 기술된 내용은 단순히 본 실시예에 대한 배경 정보를 제공할 뿐 종래기술을 구성하는 것은 아니다.

[0003] GPGPU(General Purpose Computing on Graphics Processing Units)는 다수의 스레드에서 동시에 연산을 진행하며, 이를 위해 각각의 스레드에 레지스터를 할당해야 한다. 대용량 데이터를 처리하기 위해 코어가 증가하면서 GPGPU의 레지스터 파일 크기는 갈수록 커지는 추세이다. 현재 레지스터 파일은 SRAM으로 구성되어 있으며 전체 GPGPU 에너지에서 15 내지 20 %를 차지한다.

[0004] 레지스터 파일이 사용하는 에너지의 평균 50.6%가 누설 전류로 인해 소모되는 실정이다. GPGPU의 레지스터 파일 크기가 증가함에 따라 에너지 사용을 감소시키는 방안이 필요하다.

선행기술문헌

특허문헌

[0005] (특허문헌 0001) 한국공개공보 제10-2016-0138878호 (2016.12.06)

발명의 내용

해결하려는 과제

[0006] 본 발명의 실시예들은 GPGPU(General Purpose Computing on Graphics Processing Units)의 연산시 사용되는 레지스터 파일에 STT-RAM(Spin Transfer Torque-Random Access Memory)을 사용하여 에너지 효율을 높이는 데 발명의 주된 목적이 있다.

[0007] 본 발명의 명시되지 않은 또 다른 목적들은 하기의 상세한 설명 및 그 효과로부터 용이하게 추론할 수 있는 범위 내에서 추가적으로 고려될 수 있다.

과제의 해결 수단

[0008] 본 실시예의 일 측면에 의하면, 연산자와 피연산자를 포함하는 명령어의 집합인 데이터 처리 단위를 선택하는 스케줄러, 상기 피연산자에 대응하는 데이터를 계층적으로 저장하는 저장부, 상기 저장부로부터 상기 데이터를 수신하여 상기 피연산자의 준비 상태를 확인하는 피연산자 수집기, 상기 피연산자의 준비 상태에 따라 상기 데이터 처리 단위를 실행하는 연산부, 및 상기 연산부로부터 수신한 접근 요청을 저장하고 저장된 접근 요청을 상

기 저장부에 할당하는 중계기를 포함하는 레지스터 파일 장치를 제공한다.

[0009] 본 실시예의 다른 측면에 의하면, 그래픽 처리 장치에 있어서, 데이터를 병렬 처리하는 멀티 프로세서, 및 상기 데이터를 저장하는 레지스터 파일 장치를 포함하며, 상기 레지스터 파일 장치는, 연산자와 피연산자를 포함하는 명령어의 집합인 데이터 처리 단위를 선택하는 스케줄러, 상기 피연산자에 대응하는 데이터를 계층적으로 저장하는 저장부, 상기 저장부로부터 상기 데이터를 수신하여 상기 피연산자의 준비 상태를 확인하는 피연산자 수집기, 상기 피연산자의 준비 상태에 따라 상기 데이터 처리 단위를 실행하는 연산부, 및 상기 연산부로부터 수신한 접근 요청을 저장하고 저장된 접근 요청을 상기 저장부에 할당하는 중계기를 포함하는 것을 특징으로 하는 그래픽 처리 장치를 제공한다.

발명의 효과

[0010] 이상에서 설명한 바와 같이 본 발명의 실시예들에 의하면, GPGPU의 연산시 사용되는 레지스터 파일에 STT-RAM을 사용하여 에너지 효율을 높이고, STT-RAM와 함께 레지스터 캐시와 버퍼를 계층적으로 사용함으로써, 누설 전류를 최소화하면서 쓰기 동작 전력을 줄이고 쓰기 지연을 해소할 수 있는 효과가 있다.

[0011] 여기에서 명시적으로 언급되지 않은 효과라 하더라도, 본 발명의 기술적 특징에 의해 기대되는 이하의 명세서에서 기재된 효과 및 그 잠정적인 효과는 본 발명의 명세서에 기재된 것과 같이 취급된다.

도면의 간단한 설명

[0012] 도 1은 기존의 SRAM을 이용한 레지스터 파일 장치를 예시한 블록도이다.

도 3 및 도 4는 본 발명의 실시예들에 따른 STT-RAM을 이용한 레지스터 파일 장치를 예시한 블록도이다.

도 4 내지 도 6는 본 발명의 실시예들에 따른 STT-RAM을 이용한 레지스터 파일 장치의 동작을 예시한 흐름도이다.

도 7은 본 발명의 실시예들에 따른 STT-RAM을 이용한 레지스터 파일 장치에 적용된 압축 방식을 예시한 도면이다.

발명을 실시하기 위한 구체적인 내용

[0013] 이하, 본 발명을 설명함에 있어서 관련된 공지기능에 대하여 이 분야의 기술자에게 자명한 사항으로서 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명을 생략하고, 본 발명의 일부 실시예들을 예시적인 도면을 통해 상세하게 설명한다.

[0014] 도 1은 기존의 SRAM을 이용한 레지스터 파일 장치를 예시한 블록도이다.

[0015] 워프 스케줄러(Warp Scheduler)는 어떤 워프(Warp)를 실행시킬지 스케줄링하는 유닛이다. 데이터 처리 단위는 스레드(Thread), 워프(Warp), 블록(Bloack), 그리드(Grid) 등으로 구분될 수 있다. 워프는 복수의 스레드가 묶인 그룹이고, 블록은 복수의 워프가 묶인 그룹이고, 그리드는 복수의 블록이 묶인 그룹이다.

[0016] GPGPU의 스트리밍 프로세서(Stream Processor, SP)는 복수의 스레드를 실행한다. 동일 스트림 프로세서들에 할당된 스레드들은 공유 메모리(Shared Memory)를 공유하여 사용할 수 있고, 동일 스트림 프로세서들에 할당된 스레드들은 상수 캐시(Constant Cache) 및 텍스처 캐시(Texture Cache)에 접근할 수 있다. 각 스레드를 실행하기 위한 특정 개수의 클럭이 설정되고, 특정 개수의 클럭으로 설정된 사이클 단위마다 복수의 스레드가 실행될 수 있다. 스트리밍 멀티 프로세서는 워프를 실행한다.

[0017] 디바이스 메모리(Device Memory)에 스트리밍 멀티프로세서(Streaming Multiprocessor, SM)들이 연결되어 있다. 각 스트리밍 멀티프로세서에는 여러 개의 스트림 프로세서(Stream Processor, SP)가 탑재된다. 블록은 멀티프로세서에서 스케줄링(Scheduling)의 기본 단위로 처음 할당된 멀티프로세서에서 실행이 완료될 때까지 계속 수행되며, 블록 간 실행 순서 및 처리는 상호 독립적이다. 실행 준비된 다른 블록들이 사용할 충분한 공유 자원이 남아 있다면, 동시에 여러 개의 블록들이 할당되어 실행될 수 있다. 각 멀티프로세서에 할당된 블록은 워프로 나뉘어 독립적으로 실행되며, 워프 스케줄러에 의해 스케줄링된다.

[0018] 워프에 속한 스레드들은 동시에 하나의 명령어만을 실행시키기 때문에 워프의 모든 스레드들이 같은 실행 경로를 가질 때 최대 효율을 얻는다.

[0019] GPGPU는 스레드들의 효율적인 데이터 처리를 위해 다양한 종류의 메모리를 지원하며, 각 스레드는 실행되는 동

안 개별적인 레지스터(Register)와 지역 메모리(Local Memory) 공간을 할당받아 사용한다. 같은 블록 안의 스레드들은 공유 메모리(Shared Memory)를 통해 서로 통신할 수 있다.

[0020] 중계기(Arbiter)는 뱅크 충돌(Bank Conflict)이 발생하지 않게 접근 요청(Access Request)를 뱅크(bank)에 할당한다. 레지스터 파일(Register File)은 뱅크 구조로 되어 있다. 피연산자 수집기(Operand Collector)는 어떤 명령어(Instruction)의 피연산자들이 준비되었는지 알려주는 유닛이다. 연산부(Functional Unit)는 실제 연산을 수행한다.

[0021] 쓰기 동작을 설명하면, 연산부의 연산 결과로 인해 레지스터 파일에 있는 값들이 바뀌어야 한다. 예컨대, 워크 1에서 r1+r2=r3라는 코드를 실행할 때, 원래 r1=1, r2=1, r3=1이였다면 r3는 2로 바뀌게 된다. 워프 1의 r3 값이 2로 바뀌었으므로, 워프 1의 r3 값을 갱신하기 위한 쓰기 요청(Write Request)를 중계기의 큐(queue)에 저장한다. 만약 다음 명령어가 r4+r5=r6이고, 피연산자 수집기에서 모든 피연산자들이 준비되어 있으면, 연산부는 다음 명령어를 수행한다. 중계기는 큐에 있는 쓰기 요청과 읽기 요청을 고려하여 뱅크 충돌이 발생하지 않게 뱅크에 접근한다. 뱅크에 접근해 워프 1의 r3에 해당하는 값을 갱신한다.

[0022] 읽기 동작을 설명하면, 쓰기 동작과 유사하게, 연산부에서 읽기 요청(Read Request)을 보낸다. 중계기가 큐에 있는 쓰기 요청과 읽기 요청을 고려하여 뱅크에 접근한다. 뱅크에서 값을 읽은 후 피연산자 수집기에 값을 보낸다.

[0023] GPGPU의 기존의 레지스터 파일은 SRAM(Static Random Access Memory)을 이용하며, 레지스터 파일의 에너지지는 전체 에너지의 15 내지 20 %를 차지한다. SRAM이 적용된 레지스터 파일이 사용하는 에너지의 평균 50.6 %는 누설 전류로 인해 소모된다.

[0024] 본 실시예에 따른 레지스터 파일 장치는 누설 전류가 적은 STT-RAM을 적용한다. STT-RAM은 누설 전류는 매우 작지만 쓰기 전력이 SRAM 대비 10배 정도 크고, 쓰기 지연이 SRAM 대비 10배 정도이다. 이를 해결하기 위해 본 실시예에 따른 레지스터 파일 장치는 계층적 메모리 구조와 압축 방식을 적용하여 GPGPU 레지스터 파일의 에너지 효율을 높인다. SRAM를 사용한 레지스터 캐시를 구비하고, STT-RAM의 쓰기 지연을 해결하기 위해 버퍼를 구비한다.

[0025] 본 실시예에 따른 레지스터 파일 장치는 다른 실시예에 따른 그래픽 처리 장치에 적용될 수 있다.

[0026] 도 3 및 도 4는 본 발명의 실시예들에 따른 STT-RAM을 이용한 레지스터 파일 장치를 예시한 블록도이다.

[0027] 도 3에 도시한 바와 같이, 레지스터 파일 장치(10)는 스케줄러(100), 저장부(200), 중계기(300), 피연산자 수집기(400), 및 연산부(500)를 포함한다. 레지스터 파일 장치(10)는 도 3에서 예시적으로 도시한 다양한 구성요소들 중에서 일부 구성요소를 생략하거나 다른 구성요소를 추가로 포함할 수 있다. 예컨대, 압축부(600) 및 해제부(700)를 포함할 수 있다.

[0028] 스케줄러(100, 105)는 연산자와 피연산자를 포함하는 명령어의 집합인 데이터 처리 단위를 선택한다. 데이터 처리 단위는 워프를 의미할 수 있다.

[0029] 저장부(200)는 피연산자에 대응하는 데이터를 계층적으로 저장한다. 저장부(200)는 최근에 사용된 데이터를 저장하는 제1 저장부(210), 제1 저장부(220)로부터 방출된 데이터를 기 설정된 버퍼 주기 동안 저장하는 제2 저장부(230), 및 제2 저장부(220)로부터 수신한 데이터를 저장하는 제3 저장부(230)를 포함한다.

[0030] 제1 저장부(210)는 레지스터 캐시(215)로 칭하고, 제2 저장부(220)는 버퍼(225)로 칭하고, 제3 저장부(230)는 레지스터 파일(235)로 칭할 수 있다. 레지스터 캐시(215) 및 버퍼(225)는 정적 랜덤 액세스 메모리(Static Random Access Memory, SRAM)으로 구현되고, 레지스터 파일(235)은 스핀 전달 토크 랜덤 액세스 메모리(Spin Transfer Torque-Random Access Memory, STT-RAM)으로 구현될 수 있다.

[0031] 자기 메모리(Magnetic Random Access Memory, MRAM)에서는 메모리에 데이터를 쓰기 위해서 STT(Spin Transfer Torque) 현상을 이용한다. STT 현상은 스핀이 정렬된 전류가 강자성체 내를 지날 때 순간적으로 발생한 각운동량의 변화에 의하여 강자성체의 각운동량으로 전달되는 현상을 말한다. 정렬된 스핀방향을 지닌 높은 밀도의 전류가 강자성체에 입사할 경우에 강자성체의 자화 방향이 전류의 스핀 방향과 일치하지 않으면 전류의 스핀 방향으로 정렬하려는 현상을 이용하여 데이터를 쓰게 된다.

[0032] 중계기(300, 305)는 연산부(500, 505)로부터 수신한 접근 요청을 저장하고 저장된 접근 요청을 저장부에 할당한다.

[0033] 피연산자 수집기(400, 405)는 저장부로부터 데이터를 수신하여 피연산자의 준비 상태를 확인한다.

[0034] 연산부(500, 505)는 피연산자의 준비 상태에 따라 데이터 처리 단위를 실행한다.

[0035] 레지스터 파일 장치(10)는 제3 저장부(230, 235)에 저장되기 전에 기 설정된 압축 주기 동안 데이터를 압축하는 압축부(600) 및 압축된 데이터를 해제하는 해제부(700)를 추가로 포함할 수 있다.

[0036] 압축부(600)는 복수의 데이터에 대해서 기준 데이터를 설정하고 기준 데이터와의 차이로 표현하는 방식으로 데이터의 비트 수를 감소시킨다. 해제부(700)는 기준 데이터에 기준 데이터와의 차이를 더하여 복원한다.

[0037] 레지스터 파일 장치(10)는 버퍼 주기를 세고 압축 주기를 세는 카운터를 추가로 포함할 수 있다. 예컨대, 버퍼 주기는 STT-RAM이 쓰기 동작을 수행하는 데 필요한 10 사이클, 압축 주기는 압축부가 압축 동작을 수행하는 데 필요한 2 사이클로 설정될 수 있다.

[0038] 도 4 내지 도 6는 본 발명의 실시예들에 따른 STT-RAM을 이용한 레지스터 파일 장치의 동작을 예시한 흐름도이다.

[0039] 전체적인 데이터 흐름을 설명하면 가장 최근에 쓰여진 레지스터 데이터는 레지스터 캐시에 저장된다. 레지스터 캐시에서 캐시 히트(Cache Hit)가 되면 방출(Eviction)없이 레지스터 캐시에서 레지스터를 읽기/쓰기(Read/Write)를 수행한다. 레지스터 캐시에서 쓰기 미스(Write Miss)가 발생하면 데이터를 방출시키고 레지스터 캐시에 쓰기를 수행한다. 방출된 데이터는 쓰기 지연 버퍼로 넘어간다.

[0040] 쓰기 지연 버퍼에 있는 데이터는 압축부를 통해 압축 과정을 거쳐 STT-RAM 레지스터 파일에 쓰기를 수행한다. 쓰기 지연 버퍼는 STT-RAM에서 쓰기 동작에 걸리는 10 사이클과 압축 동작에 걸리는 2 사이클 동안 레지스터 데이터를 보관한다.

[0041] 레지스터 캐시에서 읽기 미스(Read Miss)가 발생하면, STT-RAM으로 이루어진 레지스터 파일에 있는 데이터를 읽기를 수행한다. 해제부를 통해 복원된 데이터가 연산부로 보내지고 GPGPU의 연산이 동작한다.

[0042] 도 5를 참조하여 쓰기 동작을 설명하면, r1=1, r2=1, r3=1일 때, 워프 1에서 r1+r2=r3라는 코드를 연산부에서 실행시키면 워프 1의 r3가 2로 레지스터 파일에 갱신되어야 한다.

[0043] 단계 S510에서 연산부는 레지스터 캐시로 쓰기 요청을 전송한다.

[0044] 단계 S520에서 접근 요청이 쓰기 요청이고 쓰기 요청에 해당하는 피연산자가 제1 저장부에 존재하는지 판단한다. 레지스터 캐시에 있는 태그를 확인한다.

[0045] 쓰기 요청에 해당하는 피연산자가 존재하면, 단계 S530에서 제1 저장부는 새로운 데이터로 덮어 쓴다. 예컨대, 캐시 히트이면 레지스터 캐시에 워프 1의 r3 값을 그대로 덮어 쓴다.

[0046] 접근 요청이 쓰기 요청이고 쓰기 요청에 해당하는 피연산자가 제1 저장부에 존재하지 않으면, 단계 S540에서 제1 저장부는 기존 데이터를 제2 저장부로 방출한 후 단계 S530에서 제1 저장부는 새로운 데이터로 덮어 쓴다. 예컨대, 캐시 미스라면 기존의 있던 데이터를 방출시킨 후, 레지스터 캐시에 워프 1의 r3 값을 갱신한다.

[0047] 접근 요청이 쓰기 요청이면, 단계 S550에서 제2 저장부는 제1 저장부로부터 방출된 데이터를 버퍼 주기 및 압축 주기 동안 저장하고, 단계 S560에서 압축부가 제2 저장부로부터 수신한 데이터를 압축한 후 단계 S570에서 제3 저장부는 압축된 데이터를 저장한다. 예컨대, 방출되어 버퍼에 저장된 데이터는 압축 과정을 거쳐 STT-RAM에 쓰기를 수행한다. STT-RAM에 쓰기 동작을 수행하는데 10 사이클, 압축이 진행되는데 2 사이클이 소요되므로, 버퍼는 카운터를 통해 12 사이클 동안 데이터를 저장한다.

[0048] 도 6를 참조하여 읽기 동작을 설명하면, 워프 1의 r3를 읽기 동작을 수행할 때, 먼저 레지스터 캐시를 확인한다.

[0049] 단계 S610에서 연산부는 레지스터 캐시로 읽기 요청을 전송한다.

[0050] 단계 S620에서 접근 요청이 읽기 요청이고 읽기 요청에 해당하는 피연산자가 제1 저장부에 존재하는지 판단한다. 읽기 요청에 해당하는 피연산자가 존재하면, 단계 S630에서 제1 저장부는 피연산자에 대응하는 데이터를 피연산자 수집기로 전송한다. 예컨대, 캐시 히트라면 그대로 피연산자 수집기로 데이터를 보낸다.

[0051] 접근 요청이 읽기 요청이고 읽기 요청에 해당하는 피연산자가 제1 저장부에 존재하지 않으면, 단계 S640에서 읽기 요청에 해당하는 피연산자가 제2 저장부에 존재하는지 확인한다. 예컨대, 캐시 미스라면 버퍼에 워프 1의 r3

가 있는지 확인한다.

[0052] 읽기 요청에 해당하는 피연산자가 제2 저장부에 존재하면, 단계 S650에서 제2 저장부는 피연산자에 대응하는 데이터를 피연산자 수집기로 전송한다. 버퍼에 워프 1의 r3의 데이터가 있는 것으로 히트되면 해당 데이터를 피연산자 수집기로 보낸다.

[0053] 접근 요청이 읽기 요청이고 읽기 요청에 해당하는 피연산자가 제2 저장부에 존재하지 않으면, 제3 저장부를 확인한다. 예컨대, 버퍼에 워프 3의 r3의 데이터가 없다면 STT-RAM 레지스터 파일을 확인한다.

[0054] 단계 S660에서 해제부는 피연산자에 대응하는 압축된 데이터를 복원한 후 단계 S670에서 제3 저장부는 복원한 데이터를 피연산자 수집기로 전송한다.

[0055] 도 7은 본 발명의 실시예들에 따른 STT-RAM을 이용한 레지스터 파일 장치에 적용된 압축 방식을 예시한 도면이다.

[0056] 레지스터 파일 장치는 압축부를 이용하여 레지스터에 쓰이는 비트 수를 최소화한다.

[0057] 압축부는 복수의 데이터에 대해서 기준 데이터를 설정하고 기준 데이터와의 차이로 표현하는 방식으로 데이터의 비트 수를 감소시킨다. STT-RAM에 쓰이는 데이터의 비트 수를 줄이기 위해 데이터를 BDI(Base Delta Immediate) 방식으로 압축한다. 도 7에 도시된 바와 같이, BDI 방식은 여러 개의 데이터가 입력될 때 데이터를 첫 번째로 들어온 데이터를 기준으로 입력되는 데이터와 첫 번째 데이터의 차로 표현하는 압축 방식이다.

[0058] 해제부는 기준 데이터에 기준 데이터와의 차이를 더하여 복원한다.

[0059] 본 실시예들에 의하면, 레지스터 캐싱을 이용한 STT-RAM 기반 계층적 레지스터 파일은 레지스터 파일의 에너지 효율을 평균적으로 14.6 % 향상시키고, 레지스터 캐싱을 이용한 STT-RAM 기반 계층적 레지스터 파일은 레지스터 파일의 크기를 24 % 감소시킴을 확인할 수 있다.

[0060] 압축 및 레지스터 캐싱을 이용한 STT-RAM 기반 계층적 레지스터 파일은 레지스터 파일의 에너지 효율을 평균적으로 25 % 향상시키고, 압축 및 레지스터 캐싱을 이용한 STT-RAM 기반 계층적 레지스터 파일은 레지스터 파일의 크기를 22 % 감소시킴을 확인할 수 있다.

[0061] 레지스터 파일 장치에 포함된 복수의 구성요소들은 상호 결합되어 적어도 하나의 모듈로 구현될 수 있다. 구성요소들은 장치 내부의 소프트웨어적인 모듈 또는 하드웨어적인 모듈을 연결하는 통신 경로에 연결되어 상호 간에 유기적으로 동작한다. 이러한 구성요소들은 하나 이상의 통신 버스 또는 신호선을 이용하여 통신한다.

[0062] 레지스터 파일 장치는 하드웨어, 펌웨어, 소프트웨어 또는 이들의 조합에 의해 로직회로 내에서 구현될 수 있고, 범용 또는 특정 목적 컴퓨터를 이용하여 구현될 수도 있다. 장치는 고정배선형(Hardwired) 기기, 필드 프로그램 가능한 게이트 어레이(Field Programmable Gate Array, FPGA), 주문형 반도체(Application Specific Integrated Circuit, ASIC) 등을 이용하여 구현될 수 있다. 또한, 장치는 하나 이상의 프로세서 및 컨트롤러를 포함한 시스템온칩(System on Chip, SoC)으로 구현될 수 있다.

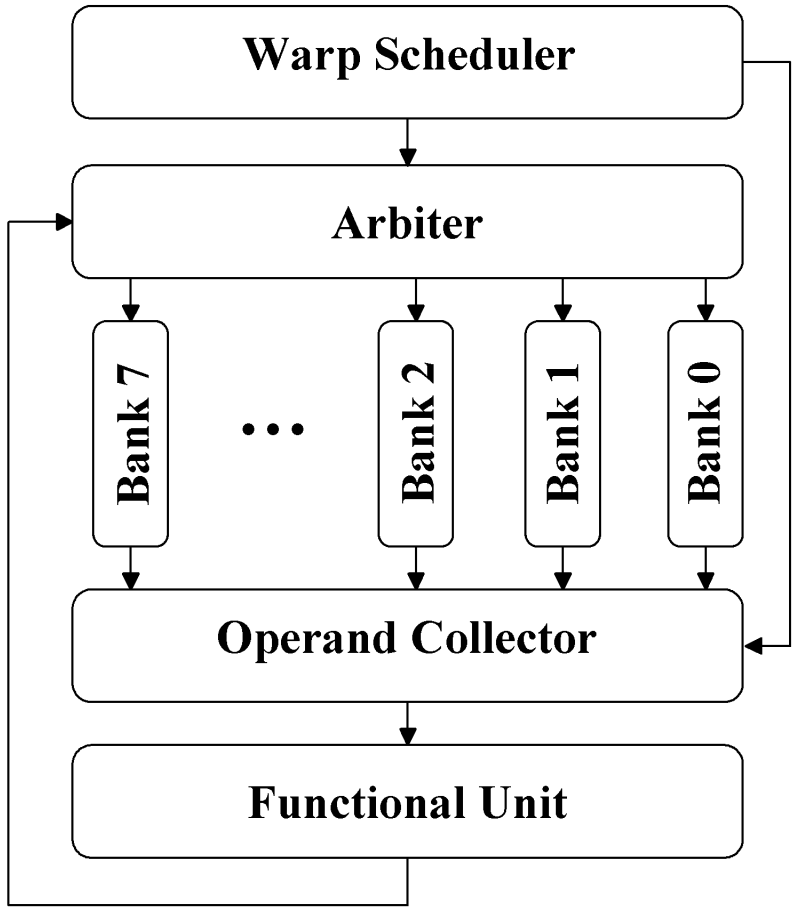
[0063] 레지스터 파일 장치는 하드웨어적 요소가 마련된 컴퓨팅 디바이스에 소프트웨어, 하드웨어, 또는 이들의 조합하는 형태로 탑재될 수 있다. 컴퓨팅 디바이스는 각종 기기 또는 유무선 통신망과 통신을 수행하기 위한 통신 모뎀 등의 통신장치, 프로그램을 실행하기 위한 데이터를 저장하는 메모리, 프로그램을 실행하여 연산 및 명령하기 위한 마이크로프로세서 등을 전부 또는 일부 포함한 다양한 장치를 의미할 수 있다.

[0064] 본 실시예들에 따른 동작은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능한 매체에 기록될 수 있다. 컴퓨터 판독 가능한 매체는 실행을 위해 프로세서에 명령어를 제공하는데 참여한 임의의 매체를 나타낸다. 컴퓨터 판독 가능한 매체는 프로그램 명령, 데이터 파일, 데이터 구조 또는 이들의 조합을 포함할 수 있다. 예를 들면, 자기 매체, 광기록 매체, 메모리 등이 있을 수 있다. 컴퓨터 프로그램은 네트워크로 연결된 컴퓨터 시스템 상에 분산되어 분산 방식으로 컴퓨터가 읽을 수 있는 코드가 저장되고 실행될 수도 있다. 본 실시예를 구현하기 위한 기능적인(Functional) 프로그램, 코드, 및 코드 세그먼트들은 본 실시예가 속하는 기술분야의 프로그래머들에 의해 용이하게 추론될 수 있을 것이다.

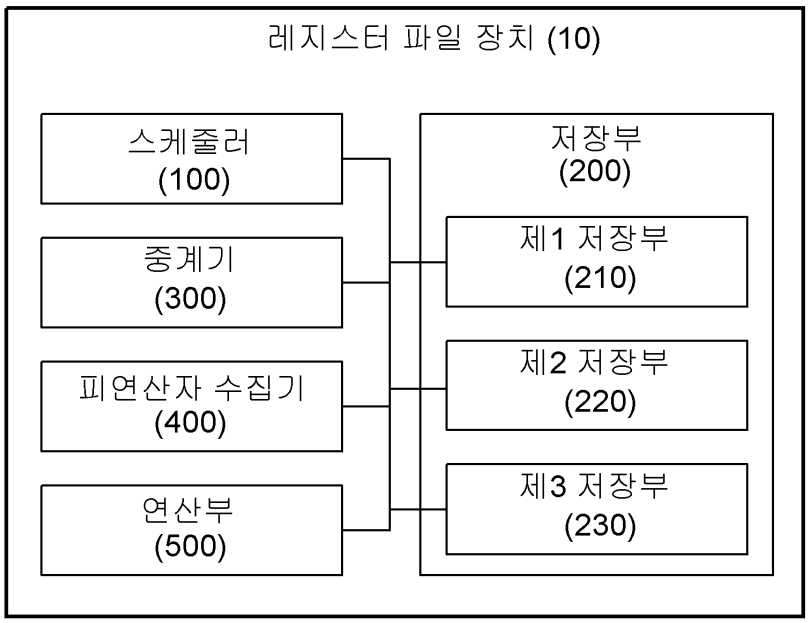
[0065] 본 실시예들은 본 실시예의 기술 사상을 설명하기 위한 것이고, 이러한 실시예에 의하여 본 실시예의 기술 사상의 범위가 한정되는 것은 아니다. 본 실시예의 보호 범위는 아래의 청구범위에 의하여 해석되어야 하며, 그와 동등한 범위 내에 있는 모든 기술 사상은 본 실시예의 권리범위에 포함되는 것으로 해석되어야 할 것이다.

도면

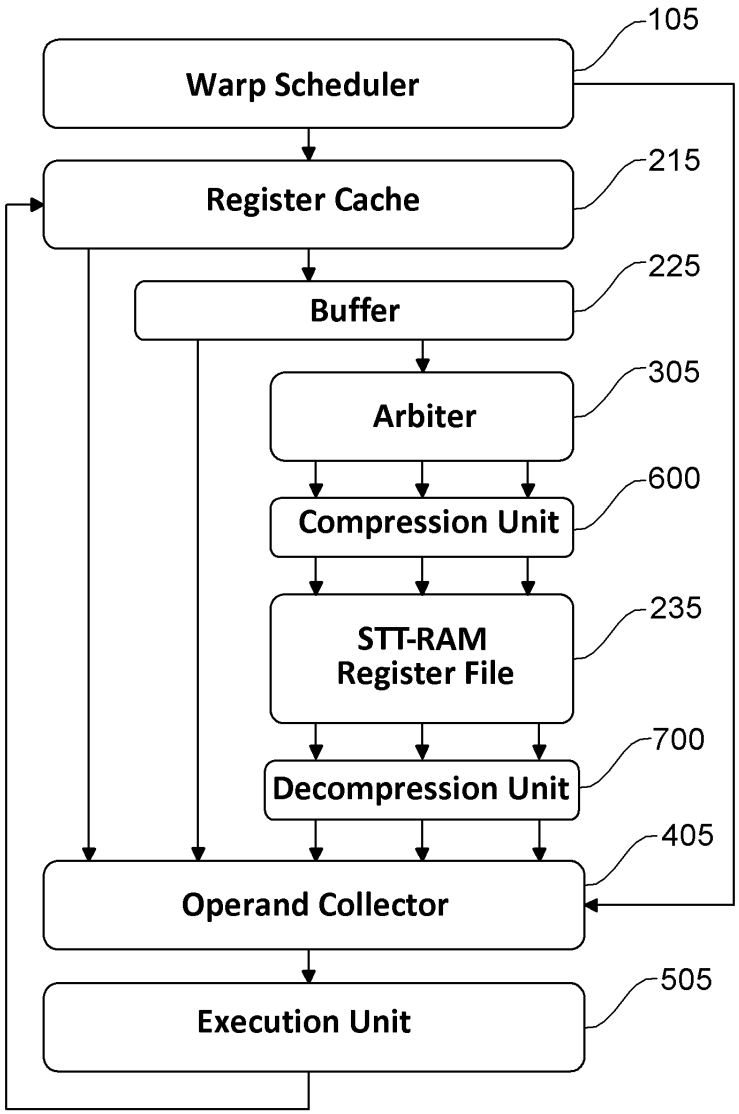
도면1


Warp Scheduler
Arbiter
Bank 7
Bank 2
Bank 1
Bank 0
Operand Collector
Functional Unit

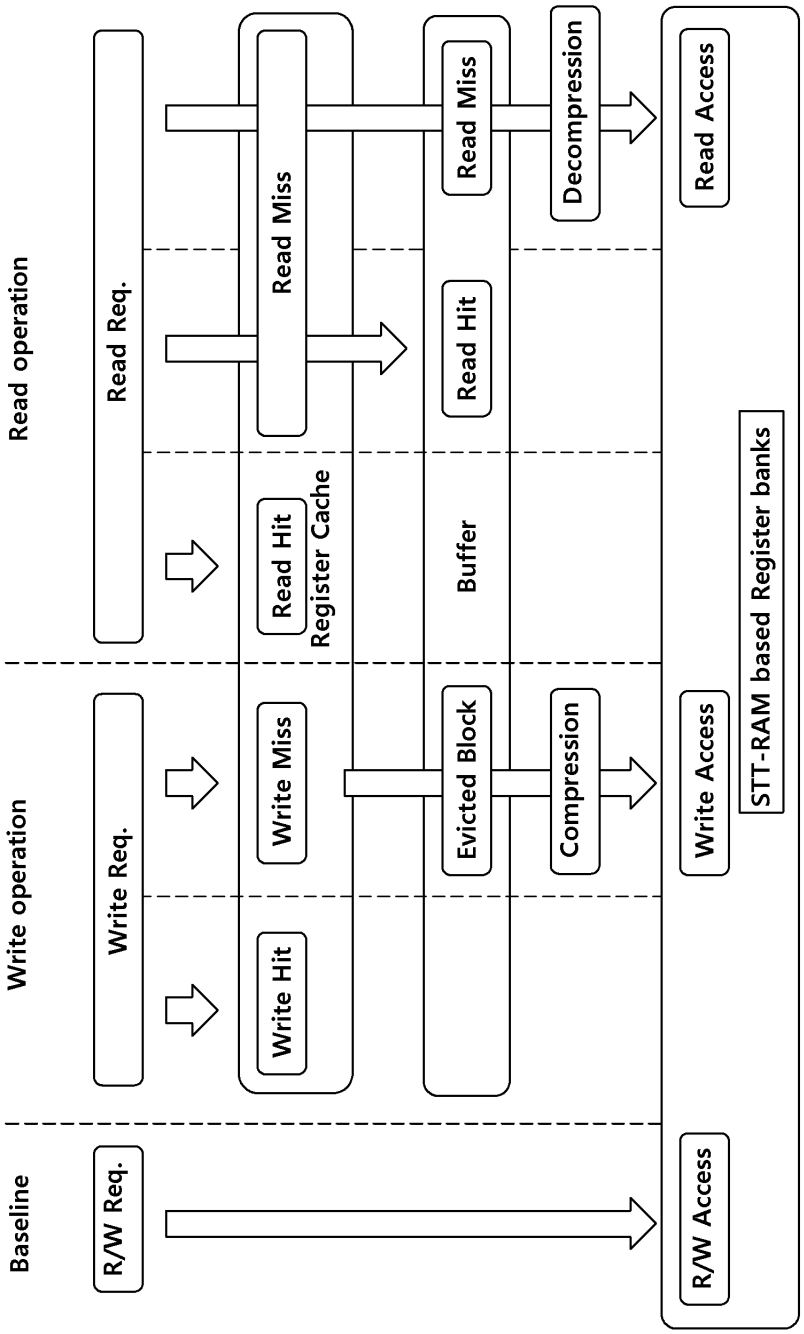
도면2


레지스터 파일 장치 (10)
스케줄러 (100)
중계기 (300)
피연산자 수집기 (400)
연산부 (500)
저장부 (200)
제1 저장부 (210)
제2 저장부 (220)
제3 저장부 (230)

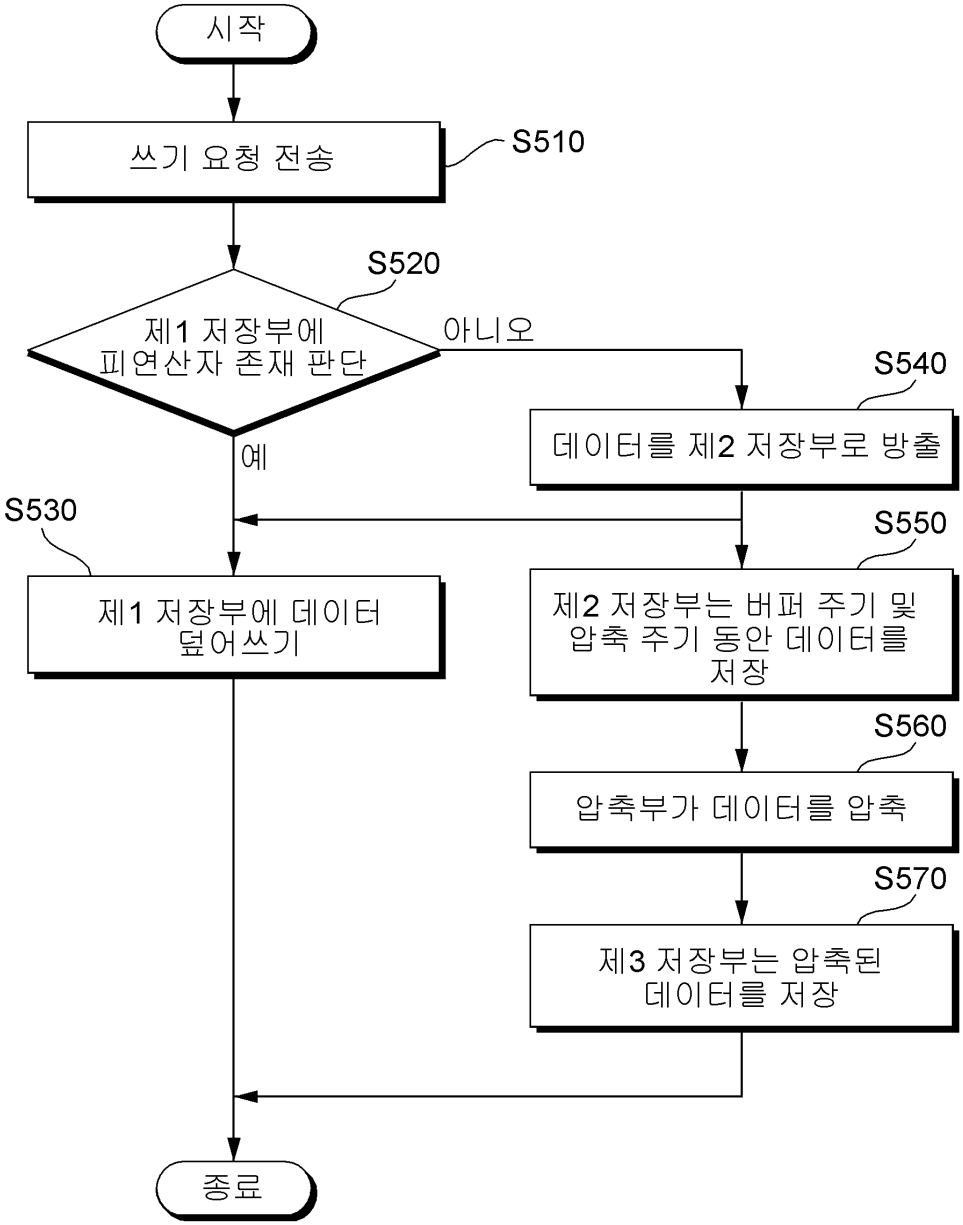
도면3


105
Warp Scheduler
215
Register Cache
225
Buffer
305
Arbiter
600
Compression Unit
235
STT-RAM
Register File
700
Decompression Unit
405
Operand Collector
505
Execution Unit

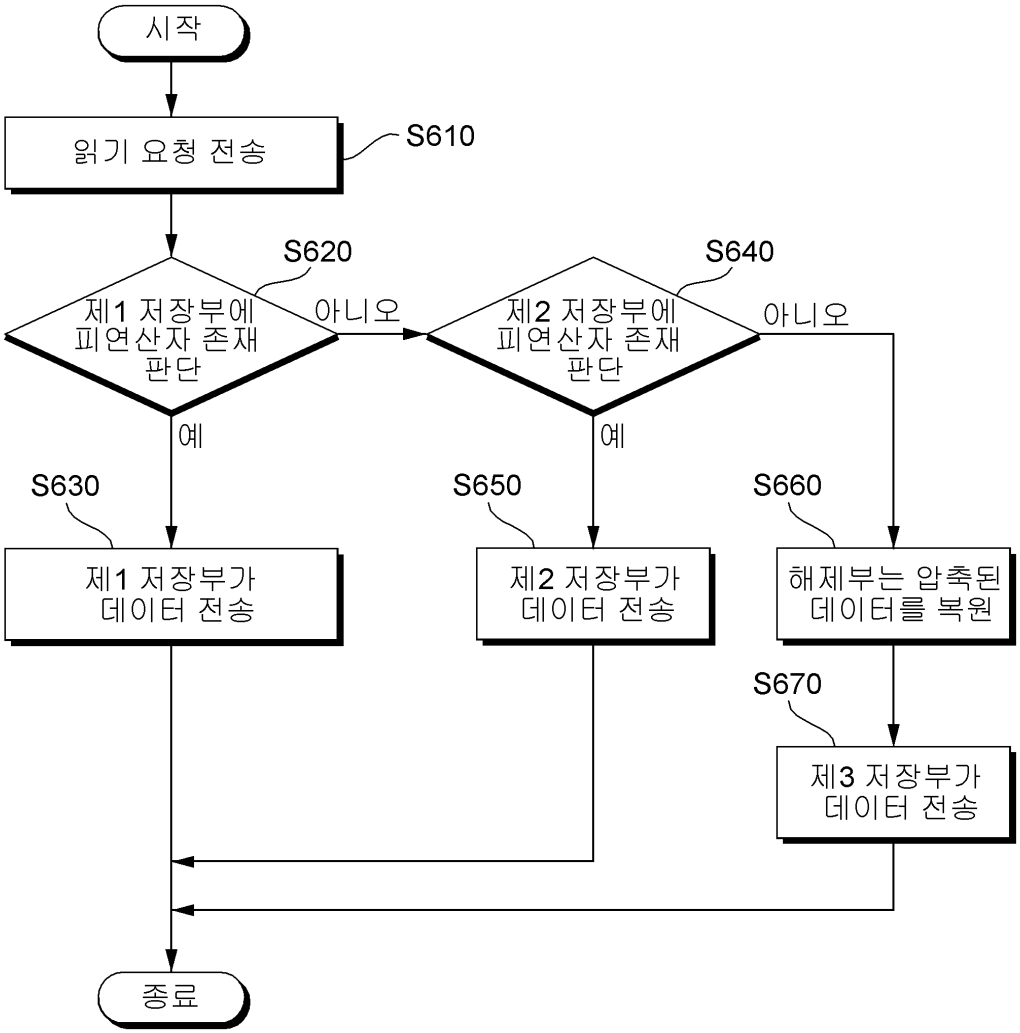
도면4


Read operation
Write operation
Baseline
Read Req.
Write Req.
R/W Req.
Read Hit
Write Hit
Read Miss
Write Miss
Register Cache
Buffer
Read Hit
Read Miss
Evicted Block
Compression
Decompression
Read Access
Write Access
R/W Access
STT-RAM based Register banks

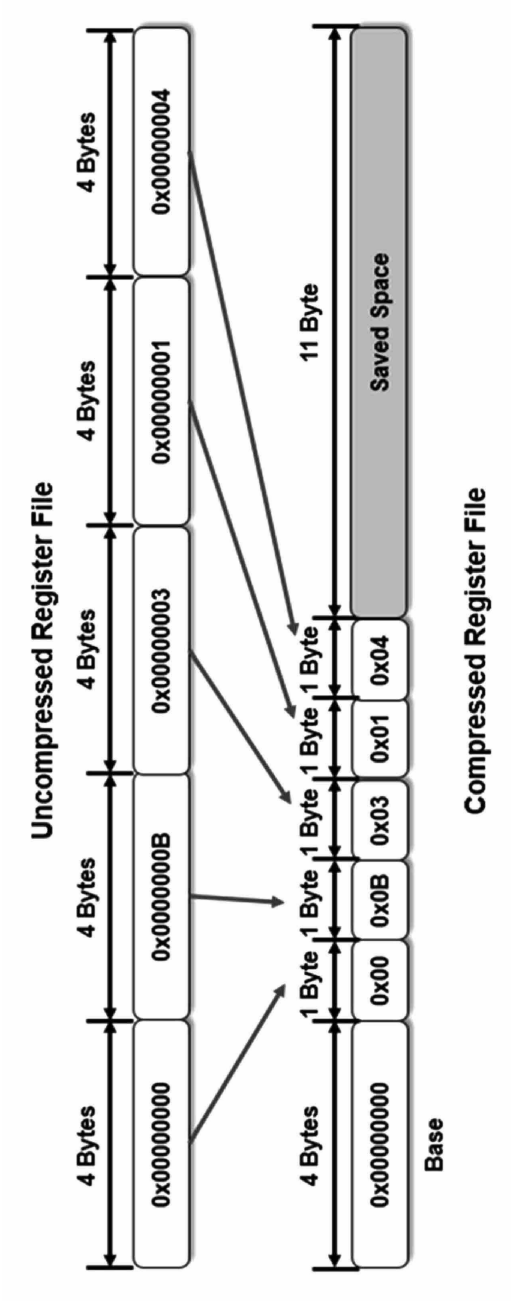
도면5


시작
쓰기 요청 전송
S510
S520
제1 저장부에
피연산자 존재 판단
아니오
예
S540
데이터를 제2 저장부로 방출
S530
제1 저장부에 데이터
덮어쓰기
S550
제2 저장부는 버퍼 주기 및
압축 주기 동안 데이터를
저장
S560
압축부가 데이터를 압축
S570
제3 저장부는 압축된
데이터를 저장
종료

도면6


시작
읽기 요청 전송
S610
S620
제1 저장부에 피연산자 존재 판단
아니오
S640
제2 저장부에 피연산자 존재 판단
아니오
예
예
S630
제1 저장부가 데이터 전송
S650
제2 저장부가 데이터 전송
S660
해제부는 압축된 데이터를 복원
S670
제3 저장부가 데이터 전송
종료

도면7


Uncompressed Register File
4 Bytes
0x00000004
4 Bytes
0x00000001
4 Bytes
0x00000003
4 Bytes
0x0000000B
4 Bytes
0x00000000
Compressed Register File
11 Byte
Saved Space
1 Byte
0x04
1 Byte
0x01
1 Byte
0x03
1 Byte
0x0B
1 Byte
0x00
4 Bytes
0x00000000
Base