



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년10월25일
(11) 등록번호 10-1789933
(24) 등록일자 2017년10월18일

(51) 국제특허분류(Int. Cl.)
G06F 3/06 (2006.01) G06F 17/30 (2006.01)
(52) CPC특허분류
G06F 3/0619 (2013.01)
G06F 17/30587 (2013.01)
(21) 출원번호 10-2016-0091395
(22) 출원일자 2016년07월19일
심사청구일자 2016년07월19일
(56) 선행기술조사문헌
KR1020130122326 A*
KR101456104 B1*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
울산과학기술원
울산광역시 울주군 언양읍 유니스트길 50
(72) 발명자
노삼혁
울산광역시 울주군 언양읍 유니스트길 50
이세권
울산광역시 울주군 언양읍 유니스트길 50
(74) 대리인
특허법인 무한
(뒷면에 계속)

전체 청구항 수 : 총 3 항

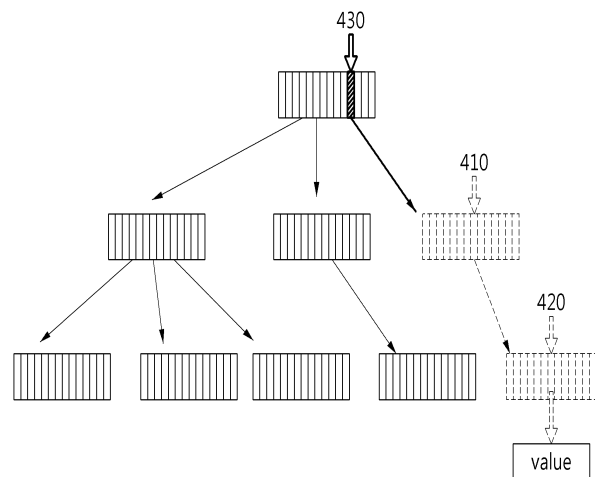
심사관 : 이철수

(54) 발명의 명칭 데이터 일관성을 위한 컴퓨팅 시스템 및 방법

(57) 요약

래디스 트리(radix tree)의 데이터 구조를 갖는 비휘발성 메모리와 캐시 플러시 호출에 대응하여 원자성 쓰기(atomic writing)를 수행하는 프로세서를 포함하는 컴퓨팅 시스템이 제공된다. 보다 구체적으로, 상기 컴퓨팅 시스템은 노드의 엔트리(entry)에 포인터를 포함하는 래디스 트리 구조로서 데이터를 저장하는 비휘발성 메모리 및 프로세스를 처리하고, 상기 프로세스 처리 과정에서 발생하는 캐시 플러시 호출에 대응하여 상기 비휘발성 메모리 내의 노드에 원자성 쓰기를 수행하는 프로세서를 포함할 수 있다.

대표도 - 도4



(52) CPC특허분류

G06F 17/30625 (2013.01)

G06F 3/0679 (2013.01)

(72) 발명자

송현섭

울산광역시 울주군 언양읍 유니스트길 50

문영제

울산광역시 울주군 언양읍 유니스트길 50

명세서

청구범위

청구항 1

프로세스를 처리하고, 프로세스 처리 과정에서 발생하는 캐시 플러시 호출에 대응하여 비휘발성 메모리 내의 노드에 원자성 쓰기(atomic writing)를 수행하는 프로세서; 및

상기 프로세서에 따라 하드웨어 적으로 원자성이 보장되는 크기에 대응하는 포인터를 노드의 엔트리(entry)에 포함하는 래덱스 트리(radix tree) 구조로서 데이터를 저장하는 비휘발성 메모리

를 포함하고,

상기 프로세서에 의해 수행되는 프로세스 내에 상기 비휘발성 메모리에 새로운 노드를 생성하는 연산이 존재하지 않는 경우, 상기 프로세서는 검출된 리프 노드에 Entry_set() 커맨드를 수행하고, 상기 리프 노드의 엔트리 영역에 sfence 커맨드에 대응하는 값 객체를 set 형태로 상기 비휘발성 메모리 영역에 전달하고, clflush를 수행하여 상기 sfence 커맨드에 대응하는 캐시 내의 주소 영역에 대해 플러시를 수행하여 해당 캐시 라인을 삭제하고,

상기 프로세서에 의해 수행되는 프로세스 내에 상기 비휘발성 메모리 내에 새로운 노드를 생성하는 연산이 존재하는 경우, 상기 프로세서는 Entry_set() 커맨드를 수행하여 새로운 리프 노드의 엔트리 영역 내에 미리 지정된 영역에 대응하는 값 객체를 set 형태로서 상기 비휘발성 메모리 영역에 전달하고, 해당 영역에 상응하는 캐시 영역에 clflush를 수행하고, 상기 새로운 리프 노드에 관한 쓰기가 완료된 경우에, 상기 래덱스 트리 구조 내에 존재하는 기생성된 노드에 관한 연결을 원자성 쓰기를 통해 생성하고,

상기 원자성이 보장되는 크기에 대응하는 포인트의 기본 단위는 8 바이트 크기의 포인터를 나타내는 컴퓨팅 시스템.

청구항 2

삭제

청구항 3

삭제

청구항 4

삭제

청구항 5

제1항에 있어서,

상기 비휘발성 메모리는,

상기 프로세서에 의해 상기 래덱스 트리 구조로서 저장된 데이터를 이용하는 데이터베이스 관리 시스템(DBMS: DataBase Management System)의 저장 장치로서 구동되는 컴퓨팅 시스템.

청구항 6

삭제

청구항 7

삭제

청구항 8

삭제

청구항 9

삭제

청구항 10

프로세스를 처리하는 프로세서에 따라 하드웨어 적으로 원자성이 보장되는 크기에 대응하는 포인터를 노드의 엔트리에 포함하는 래덱스 트리 구조로서 비휘발성 메모리에 데이터를 저장하는 단계; 및

프로세스 처리 과정에서 발생하는 캐시 플러시 호출에 대응하여 상기 래덱스 트리 내의 노드에 원자성 쓰기를 수행하는 단계

를 포함하고,

상기 원자성 쓰기를 수행하는 단계는,

상기 프로세스 내에 상기 비휘발성 메모리에 새로운 노드를 생성하는 연산이 존재하지 않는 경우, 검출된 리프 노드에 Entry_set() 커맨드를 수행하고, 상기 리프 노드의 엔트리 영역에 sfence 커맨드에 대응하는 값 객체를 set 형태로 상기 비휘발성 메모리 영역에 전달하고, clflush를 수행하여 상기 sfence 커맨드에 대응하는 캐시 내의 주소 영역에 대해 플러시를 수행하여, 해당 캐시 라인을 삭제하는 단계; 및

상기 프로세스 내에 상기 비휘발성 메모리에 새로운 노드를 생성하는 연산이 존재하는 경우, Entry_set() 커맨드를 수행하여 새로운 리프 노드의 엔트리 영역 내에 미리 지정된 영역에 대응하는 값 객체를 set 형태로서 상기 비휘발성 메모리 영역에 전달하고, 해당 영역에 상응하는 캐시 영역에 clflush를 수행하고, 상기 새로운 리프 노드에 관한 쓰기가 완료된 경우에, 상기 래덱스 트리 구조 내에 존재하는 기생성된 노드에 관한 연결을 원자성 쓰기를 통해 생성하는 단계

를 더 포함하고,

상기 원자성이 보장되는 크기에 대응하는 포인터의 기본 단위는 8 바이트 크기의 포인터를 나타내는 비휘발성 메모리를 포함하는 데이터베이스 시스템의 관리 방법.

청구항 11

삭제

발명의 설명

기술 분야

[0001] 컴퓨팅 시스템 및 컴퓨팅 방법에 연관되며, 보다 특정하게는 데이터 일관성을 보장하는 컴퓨팅 시스템 및 방법에 연관된다.

배경 기술

[0002] 컴퓨팅 시스템에 이용되는 휘발성 램(RAM: Random Access Memory)은 시스템의 붕괴 시 데이터를 소실한다. 따라서, 데이터 갱신에 이용되는 연산 처리의 일관성이나 원자성(atomicity)의 보장이 큰 이슈가 되지 않는다.

[0003] 반면, 비휘발성 메모리(NVRAM: Non-volatile Random Access Memory)를 데이터 저장 장치의 용도로 사용하는 경우에는 시스템 붕괴 시에도 데이터가 소실되지 않는데, 소실되지 않은 데이터의 재사용을 위해서는 상기 데이터의 일관성 또는 원자성이 보장되어야 한다.

[0004] 이를 테면 중앙처리장치(CPU: Central Processing Unit)의 캐시 플러시 호출에 대응하여 비휘발성 메모리에 저장되는 데이터의 일관성이 유지될 필요가 존재한다. 종래에는 비휘발성 메모리 내의 데이터 구조를 B+ 트리(B+ tree) 구조로서 구현하고, 로깅 영역(logging area)에 새롭게 저장될 데이터를 미리 저장하고 캐시 플러시 호출

에 대한 연산을 수행하는 로깅 방법이나 엔트리(entry)에 포함되는 데이터를 복사하여 업데이트한 이후에 포인터를 바꾸는 방식의 섀도잉(shadowing) 방법이 그 해결 방안으로 이용되었다. 다만, B+ 트리 구조는 엔트리 내의 키(key)와 포인터의 크기가 원자성이 유지되는 8 byte를 초과한다는 점에서 갑작스러운 크래쉬(crash)가 발생하는 경우에 데이터의 일관성이 깨지게 될 가능성이 존재하여 그 새로운 해결 방안이 필요한 실정이다.

발명의 내용

과제의 해결 수단

- [0005] 일측에 따르면, 래덱스 트리(radix tree)의 데이터 구조를 갖는 비휘발성 메모리와 캐시 플러시 호출에 대응하여 원자성 쓰기(atomic writing)를 수행하는 프로세서를 포함하는 컴퓨팅 시스템이 제공된다. 보다 구체적으로, 상기 컴퓨팅 시스템은 노드의 엔트리(entry)에 포인터를 포함하는 래덱스 트리 구조로서 데이터를 저장하는 비휘발성 메모리 및 프로세스를 처리하고, 상기 프로세스 처리 과정에서 발생하는 캐시 플러시 호출에 대응하여 상기 비휘발성 메모리 내의 노드에 원자성 쓰기를 수행하는 프로세서를 포함할 수 있다.
- [0006] 일실시예에 따르면, 상기 프로세서는 상기 포인터의 미리 지정된 크기에 대응하는 원자성 쓰기를 수행하여, 데이터의 일관성을 유지하도록 하는 것을 특징으로 할 수 있다. 보다 구체적으로, 상기 프로세서는 상기 캐시 플러시 호출에 대응하여 8 바이트(bytes)의 원자성 쓰기를 수행하는 것을 특징으로 할 수 있다.
- [0007] 다른 일실시예에 따르면, 상기 프로세서는 상기 프로세스 처리 과정에서 상기 비휘발성 메모리에 새로운 노드를 생성하는 연산이 존재하지 않는 경우에, 상기 캐시 플러시 호출에 대응하는 리프 노드에 원자성 쓰기를 수행할 수 있다.
- [0008] 또 다른 일실시예에 따르면, 상기 비휘발성 메모리는 상기 프로세서에 의해 상기 래덱스 트리 구조로서 저장된 데이터를 이용하는 데이터베이스 관리 시스템(DBMS: DataBase Management System)의 저장 장치로서 구동될 수 있다.
- [0009] 다른 일측에 따르면, 새로운 노드의 생성 연산이 존재하는 경우에 래덱스 트리 구조 내에서 루트 노드에 연결될 리프 노드를 우선적으로 생성하는 컴퓨팅 시스템이 제공된다. 보다 구체적으로, 상기 컴퓨팅 시스템은 노드의 엔트리에 포인터를 포함하는 래덱스 트리 구조로서 데이터를 저장하는 비휘발성 메모리 및 프로세스를 처리하고, 상기 프로세스 처리 과정에서 새로운 노드 생성 연산이 존재하는 경우에 미리 생성된 상기 래덱스 트리 구조 내의 엔트리 중 어느 하나에 연결될 적어도 하나의 새로운 리프 노드를 생성하는 프로세서를 포함할 수 있다.
- [0010] 일실시예에 따르면, 상기 프로세서는 상기 적어도 하나의 새로운 리프 노드의 생성이 완료된 경우에 상기 프로세스 처리 과정에서 발생하는 캐시 플러시 호출에 대응하여 상기 래덱스 트리 구조 내의 엔트리 중 어느 하나에 원자성 쓰기를 수행하고, 상기 래덱스 트리 구조 내의 엔트리와 상기 적어도 하나의 새로운 리프 노드를 연결할 수 있다.
- [0011] 다른 일실시예에 따르면, 상기 프로세서는 상기 프로세스 처리 과정에서 발생하는 캐시 플러시 호출에 대응하여 상기 새로운 리프 노드에 원자성 쓰기를 각각 수행할 수 있다.
- [0012] 또 다른 일측에 따르면, 비휘발성 메모리를 포함하는 데이터베이스 시스템의 관리 방법이 제공된다. 상기 관리 방법은 노드의 엔트리에 포인터를 포함하는 래덱스 트리 구조로서 비휘발성 메모리에 데이터를 저장하는 단계 및 프로세스 처리 과정에서 발생하는 캐시 플러시 호출에 대응하여 상기 래덱스 트리 내의 노드에 원자성 쓰기를 수행하는 단계를 포함할 수 있다. 보다 구체적으로, 상기 원자성 쓰기를 수행하는 단계는 상기 포인터의 미리 지정된 크기에 대응하는 원자성 쓰기를 수행하여, 데이터의 일관성을 유지하도록 하는 것을 특징으로 할 수 있다.

도면의 간단한 설명

- [0013] 도 1은 비휘발성 메모리가 이용되는 컴퓨팅 시스템을 나타내는 예시도이다.
- 도 2는 일실시예에 따른 래덱스 구조를 설명하는 예시도이다.
- 도 3은 일실시예에 따라 비휘발성 메모리 내에 원자성 쓰기가 수행되는 과정을 설명하는 예시도이다.
- 도 4는 다른 일실시예에 따라 비휘발성 메모리 내에 원자성 쓰기가 수행되는 과정을 설명하는 예시도이다.

도 5는 일실시예에 따른 컴퓨팅 시스템의 블록도를 나타낸다.

도 6은 일실시예에 따른 비휘발성 메모리를 포함하는 데이터베이스 시스템의 관리 방법의 흐름도를 나타낸다.

발명을 실시하기 위한 구체적인 내용

- [0014] 실시예들에 대한 특정한 구조적 또는 기능적 설명들은 단지 예시를 위한 목적으로 개시된 것으로서, 다양한 형태로 변경되어 실시될 수 있다. 따라서, 실시예들은 특정한 개시형태로 한정되는 것이 아니며, 본 명세서의 범위는 기술적 사상에 포함되는 변경, 균등물, 또는 대체물을 포함한다.
- [0015] 제1 또는 제2 등의 용어를 다양한 구성요소들을 설명하는데 사용될 수 있지만, 이런 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 해석되어야 한다. 예를 들어, 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소는 제1 구성요소로도 명명될 수 있다.
- [0016] 어떤 구성요소가 다른 구성요소에 "연결되어" 있다고 언급된 때에는, 그 다른 구성요소에 직접적으로 연결되어 있거나 또는 접속되어 있을 수도 있지만, 중간에 다른 구성요소가 존재할 수도 있다고 이해되어야 할 것이다.
- [0017] 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 명세서에서, "포함하다" 또는 "가지다" 등의 용어는 기재된 특징, 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것이 존재함으로 지정하려는 것이지, 하나 또는 그 이상의 다른 특징들이나 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.
- [0018] 다르게 정의되지 않는 한, 기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 모든 용어들은 해당 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가진다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥상 가지는 의미와 일치하는 의미를 갖는 것으로 해석되어야 하며, 본 명세서에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다.
- [0019] 이하, 실시예들을 첨부된 도면들을 참조하여 상세하게 설명한다. 첨부 도면을 참조하여 설명함에 있어, 도면 부호에 관계없이 동일한 구성 요소는 동일한 참조 부호를 부여하고, 이에 대한 중복되는 설명은 생략하기로 한다.
- [0020] 도 1은 비휘발성 메모리가 이용되는 컴퓨팅 시스템을 나타내는 예시도이다.
- [0021] 도 1을 참조하면, 비휘발성 메모리가 이용되는 컴퓨팅 시스템이 도시된다. 컴퓨팅 시스템은 중앙처리장치(또는 CPU)의 캐시 및 메모리 컨트롤러(memory controller)를 포함하는 휘발성 영역(volatile area)(110)과 Persistent memory(또는 Non-volatile memory 또는 NVRAM 또는 비휘발성 메모리)를 포함하는 비휘발성 영역(Non-volatile area)(120)으로 분리될 수 있다. 구체적으로, 휘발성 영역(110)내의 중앙처리장치는 캐시(caches)에 처리된 데이터를 저장하거나 버퍼시킬 수 있다. 또한, 중앙처리장치는 캐시에 저장된 데이터를 메모리 컨트롤러를 통하여 비휘발성 메모리로 쓰기(writing)를 할 수 있다.
- [0022] 본 실시예에서 설명되는 비휘발성 메모리의 종류에는 이를 테면, PCRAM(Phase change RAM), MRAM(Magnetoresistive RAM), FeRAM(Ferroelectric RAM) 등이 존재할 수 있다. 다만, 이러한 비휘발성 메모리의 종류 또한 본 발명의 일부 실시예에 불과하며, 본 발명이 이러한 실시예에 의해 제한적으로 해석되어서는 안 될 것이다.
- [0023] 중앙처리장치의 캐시 데이터가 휘발성 영역(110)에 포함된다는 점을 고려하면, 비휘발성 메모리를 저장 장치로서 이용하기 위해서는 캐시 데이터와의 일관성 및 신뢰성을 보장하기 위한 방법이 필요할 것이다. 보다 구체적으로, 데이터베이스 내에서 수행되는 일련의 연산들(operations)은 트랜잭션(transaction)으로 정의될 수 있다. 컴퓨팅 시스템 내에서 트랜잭션의 모드 연산들은 전체로서 완료되거나 전체로서 취소되어야 하는 원자성을 가지고 있다. 그에 따라, 휘발성 영역(110)에 존재하는 중앙처리장치로부터 clflush 커맨드가 요청된 경우에 캐시 데이터가 전체로서 비휘발성 영역(120)의 비휘발성 메모리에 기록될 필요가 존재한다. 보다 구체적으로, clflush는 오퍼랜드로 입력된 메모리 주소의 캐시 라인을 삭제하는 동작으로서, 상기 캐시 라인의 상태를 dirty 상태로 만들어 캐시 라인이 부족한 경우에 지워질 수 있는 플래그를 설정하는 연산을 나타낼 수 있다.
- [0024] 또한, 컴퓨팅 시스템에 있어서 쓰기 연산이 많이 수행될수록 전체 시스템의 성능(performance)이 크게 저하된다는 것은 기술 분야의 전문가에게는 straight forward한 사실일 것이다.

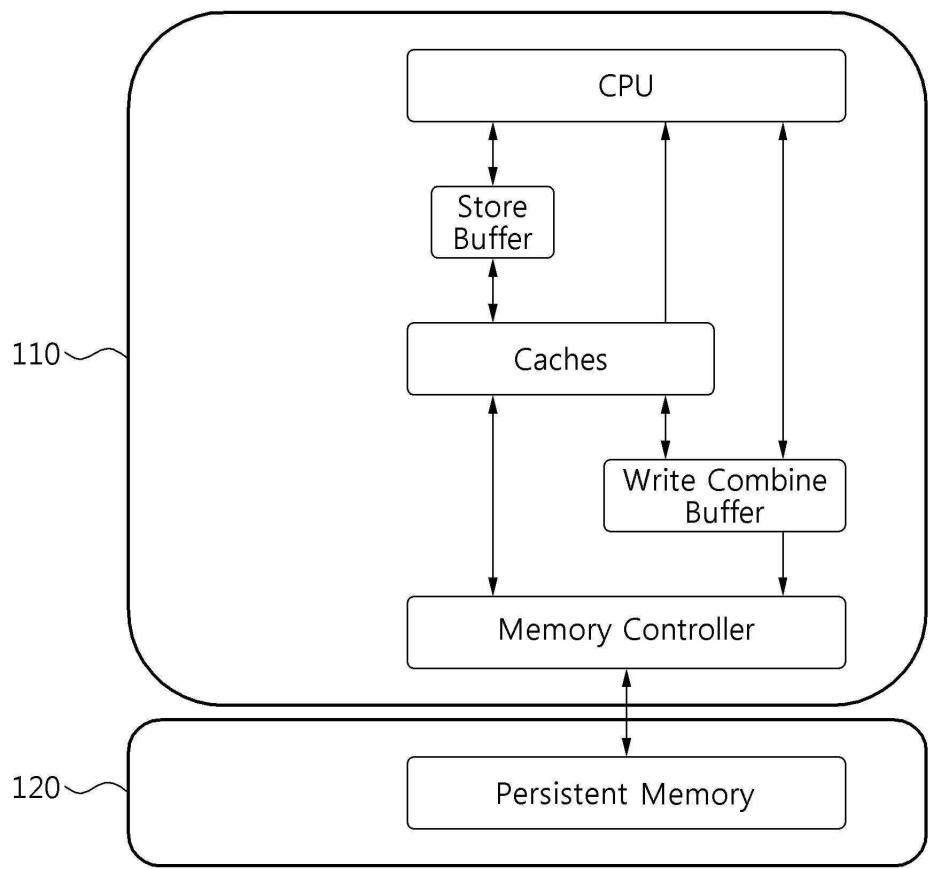
- [0025] 아래에 추가될 도면을 통하여, 위와 같이 휘발성 영역(110)과 비휘발성 영역(120)을 포함하는 컴퓨팅 시스템이 데이터의 일관성을 보장하기 위해 `clflush` 커맨드를 수행하는 구체적인 실시예들에 관한 설명이 제공될 것이다.
- [0026] 도 2는 일실시예에 따른 래딕스 구조를 설명하는 예시도이다.
- [0027] 도 2를 참조하면, 비휘발성 메모리 내에 구현된 래딕스 구조의 루트 노드(root node)(210), 내부 노드(internal node)(220) 및 리프 노드(leaf node)(230)가 도시된다.
- [0028] 래딕스 구조의 노드들(210, 220, 230)은 소정 크기에 대응하는 포인터만을 엔트리(entry)에 포함할 수 있다. 보다 구체적으로, 래딕스 구조의 노드(210, 220, 230)는 8 바이트(bytes) 크기의 포인터만을 엔트리에 포함할 수 있다. 종래의 비휘발성 메모리의 데이터 구조로서 이용되던 B+ 트리의 노드들은 아래에서 추가될 설명과 같이 키(key)와 포인터를 엔트리에 포함하여 그 크기가 8 바이트를 초과하게 된다. 그에 따라, 종래의 B+ 트리의 구조에는 8 바이트를 이용한 원자성 쓰기가 불가능한 한계점이 존재한다. 그에 따라, 종래 방식을 이용하는 경우에는 추가적인 `clflushing` 요청이 필요할 것으로 예상되어 이는 시스템 전체의 성능을 저하하는 요인이 될 수 있다.
- [0029] 따라서, 본 실시예에 따르는 경우에 종래의 방식과는 다르게 `clflushing` 요청에 상응하는 데이터의 쓰기가 수행되는 경우, 원자성 쓰기를 이용하여 데이터의 원자성 및 일관성을 유지하는 효과를 기대할 수 있을 것이다.
- [0030] 또한, 래딕스 구조의 노드들(210, 220, 230)은 종래의 B+ 트리에 포함되는 노드 엔트리와 비교할 때, 키(key)를 포함하지 않는 특징이 존재한다. 예시적으로, 종래의 NV 트리의 엔트리는 플래그(flag), 키 및 값(value)을 포함하고, wB+ 트리의 엔트리는 키와 값을 포함할 수 있다.
- [0031] 위와 같은 차이점에 기초하여 래딕스 구조의 노드들(210, 220, 230)은 루트 노드(210)로부터 내부 노드(220)를 거쳐 리프 노드(230)로 연결되는 과정에서 종래의 B+ 트리와는 다르게 키 소팅(key sorting)이나 리밸런싱(rebalancing) 과정을 수행하지 않을 수 있다.
- [0032] 도 3은 일실시예에 따라 비휘발성 메모리 내에 원자성 쓰기가 수행되는 과정을 설명하는 예시도이다.
- [0033] 도 3을 참조하면, 중앙처리장치(또는 프로세서)의 캐시 플러시 호출에 대응하여 비휘발성 메모리 내의 리프 노드(310)에 원자성 쓰기가 수행되는 과정이 도시된다. 보다 구체적으로, 본 실시예에서는 프로세서에 의해 수행되는 프로세스에 비휘발성 메모리 내에 새로운 노드를 생성하는 연산이 존재하지 않는 경우가 설명될 수 있다.
- [0034] 앞서 설명한 것과 같이, 비휘발성 메모리는 래딕스 구조로서 내부 데이터 구조가 구현될 수 있다. 래딕스 구조는 종래의 데이터 구조와는 다르게 별도의 키 소팅이나 리밸런싱 연산 없이 캐시 데이터에 상응하는 리프 노드(310)가 검출될 수 있다.
- [0035] 프로세서는 `sfence` 커맨드를 수행할 수 있다. 그에 따라, 미리 지정된 fence에 상응하는 데이터가 캐시 내에 저장될 수 있다. 또한, 프로세서는 검출된 리프 노드(310)에 `Entry_set()` 커맨드를 수행할 수 있다. 프로세서는 리프 노드(310)의 엔트리 영역에 `sfence` 커맨드에 대응하는 값(value) 객체를 set 형태로서 비휘발성 메모리 영역에 전달할 수 있다.
- [0036] 또한, 프로세서는 `clflush`를 수행할 수 있다. `sfence`에 대응하는 캐시 내의 주소 영역에 대해 플러시를 수행하고, 해당 캐시 라인을 삭제할 수 있다. 이어서, 프로세서는 다시 `sfence` 커맨드를 수행하여 fence에 상응하는 새로운 데이터가 캐시 내에 저장되어 캐시 플러시 절차를 수행할 수 있다.
- [0037] 본 실시예에 따를 때, 리프 노드(310)는 래딕스 구조의 하나의 노드로서 해당 엔트리 영역이 8 바이트 이내로 구현될 수 있다. 그에 따라, 캐시 플러시 요청에 상응하는 8 바이트 원자성 쓰기가 리프 노드(310)에 수행될 수 있고, 그 효과에 따라 전달되는 데이터의 일관성과 원자성이 보장되는 효과를 기대할 수 있을 것이다.
- [0038] 도 4는 다른 일실시예에 따라 비휘발성 메모리 내에 원자성 쓰기가 수행되는 과정을 설명하는 예시도이다.
- [0039] 도 4를 참조하면, 프로세서의 캐시 플러시 호출에 대응하여 비휘발성 메모리 내의 새로운 노드(410, 420)들이 생성되고, 미리 존재하는 래딕스 구조 내의 루트 노드(430)에 연결되어 원자성 쓰기가 수행되는 과정이 도시된다. 보다 구체적으로, 본 실시예에서는 프로세서에 의해 수행되는 프로세스에 새로운 노드들을 생성하는 연산이 존재하는 경우가 설명될 수 있다. 본 실시예에서는, 새로운 노드(410, 420)들이 미리 존재하는 래딕스 구조 내의 루트 노드(430)에 연결되는 실시예가 설명되지만, 상기 래딕스 구조 내의 엔트리 중 어느 하나의 임의의 엔트리로 연결되는 실시예로 확장 가능하다는 것은 기술 분야의 전문가에게는 자명한 사실일 것이다.

- [0040] 앞서 설명된 바와 같이, 본 실시예에 대한 비휘발성 메모리는 래딕스 구조로서 내부 데이터 구조가 구현될 수 있다. 래딕스 구조에 관한 설명은 앞서 기재한 설명들과 중복되기 때문에 자세한 설명은 생략하기로 한다.
- [0041] 프로세서는 새로운 내부 노드(internal node)(410)를 생성하기 위해 Entry_set() 커맨드를 수행할 수 있다. 프로세서는 새로운 내부 노드(410)의 엔트리 영역 내에 미리 지정된 영역에 대응하는 값(value) 객체를 set 형태로 비휘발성 메모리 영역에 전달할 수 있다. 또한, 프로세서는 해당 영역에 상응하는 캐시 영역에 clflush를 수행할 수 있다.
- [0042] 마찬가지로, 프로세서는 새로운 리프 노드(leaf node)(420)를 생성된 내부 노드(410)와 연결하기 위해 Entry_set() 커맨드를 새롭게 수행할 수 있다. 프로세서는 새로운 리프 노드(420)의 엔트리 영역 내에 미리 지정된 영역에 대응하는 값(value) 객체를 set 형태로 비휘발성 메모리 영역에 전달할 수 있다. 또한, 프로세서는 해당 영역에 상응하는 캐시 영역에 clflush를 수행할 수 있다.
- [0043] 이어서, 프로세서는 미리 존재하는 래딕스 구조의 루트 노드(430)에 관한 원자성 쓰기를 수행할 수 있다. 보다 구체적으로, 프로세서는 sfence 커맨드를 수행할 수 있다. 미리 지정된 fence에 상응하는 데이터가 캐시 내에 저장될 수 있다. 또한, 프로세서는 루트 노드(430)에 Entry_set() 커맨드를 수행할 수 있다. 이어서, 프로세서는 해당 영역에 상응하는 캐시 영역에 clflush를 수행하고, 다시 sfence 커맨드를 수행하여 fence에 상응하는 새로운 데이터가 캐시 내에 저장되어 캐시 플러시 절차를 완료할 수 있다.
- [0044] 종래의 방식에 따를 때, 프로세서에 의해 수행되는 프로세스 내에 새로운 노드의 생성 연산이 존재하는 경우에 루트 노드를 중심으로, 내부 노드, 리프 노드 순서로서 순차적으로 새로운 노드들이 연결될 수 있다. 다만, 종래의 방식의 경우에는 새로운 노드를 생성하는 과정에서 크래시가 발생하는 경우에, 컴퓨팅 시스템의 생성 중인 노드들에 대해서 롤 백(roll back) 절차를 수행할 수 없어, 새로운 노드들이 생성되기 전인 본래의 상태(original status)로 데이터 구조 내의 노드들의 연결 관계를 되돌릴 수 없다는 한계가 존재한다. 이는, 컴퓨팅 시스템 전체의 데이터 일관성을 저하하는 요인으로 작용될 수 있다.
- [0045] 본 실시예에 상응하는 딜레이드 커밋 스킴(delayed commit scheme)은 우선적으로 새롭게 생성되는 노드들(410, 420)에 관한 쓰기를 수행하고, 상기 쓰기가 완료된 경우에 본래의 루트 노드(430)에 관한 연결을 원자성 쓰기를 통해 생성할 수 있다. 그에 따라, 새롭게 생성되는 노드들(410, 420)에 관한 연산이 수행되는 도중에 크래시가 발생하더라도 본래의 래딕스 구조 자체에는 아무런 영향 없이 롤 백 절차를 수행하는 것이 가능하기 때문에 컴퓨팅 시스템 전체의 안정성을 높이고 데이터 일관성을 높이는 효과를 기대할 수 있다.
- [0046] 도 5는 일실시예에 따른 컴퓨팅 시스템의 블록도를 나타낸다.
- [0047] 도 5를 참조하면, 본 실시예에 따른 컴퓨팅 시스템(500)은 비휘발성 메모리(510) 및 프로세서(520)를 포함할 수 있다.
- [0048] 비휘발성 메모리(510)는 노드의 엔트리에 포인터를 포함하는 래딕스 트리 구조로서 데이터를 저장할 수 있다. 일실시예로서, 비휘발성 메모리(510)는 데이터베이스 관리 시스템(DBMS: DataBase Management System) 내의 저장 장치로서 구현될 수 있다.
- [0049] 프로세서(520)는 컴퓨팅 시스템(500) 전체에 연관되는 프로세스를 처리할 수 있다. 또한, 프로세서(520)는 상기 프로세스 처리 과정에서 발생하는 캐시 플러시 호출에 대응하여 상기 비휘발성 메모리 내의 노드에 원자성 쓰기를 수행할 수 있다. 프로세서(520)는 하드웨어 적으로 원자성이 보장되는 미리 지정된 크기에 대응하는 원자성 쓰기를 수행하여 데이터의 일관성을 유지할 수 있다. 보다 구체적으로, 프로세서(520)는 캐시 플러시 호출에 대응하여 8 바이트의 원자성 쓰기를 수행할 수 있다. 다만, 위에서 설명되는 8 바이트는 본 발명의 사상의 이해를 돕기 위한 예시적 기재일 뿐, 하드웨어 적으로 원자성이 보장되는 경우라면 10 바이트, 15 바이트 등과 같이 다양한 형태로 변경될 수 있다는 것은 기술 분야의 전문가에게는 자명한 사실일 것이다.
- [0050] 일실시예로서, 프로세서(520)에 의해 처리되는 프로세스 내에 비휘발성 메모리(510)에 새로운 노드를 생성하는 연산이 존재하지 않는 경우가 설명될 수 있다. 프로세서(520)는 상기 캐시 플러시 호출에 대응하는 비휘발성 메모리(510) 내의 리프 노드에 원자성 쓰기를 수행할 수 있다.
- [0051] 다른 일실시예로서, 프로세서(520)에 의해 처리되는 프로세스 내에 새로운 노드의 생성 연산이 존재하는 경우가 설명될 수 있다. 우선적으로, 프로세서(520)는 미리 존재하는 래딕스 구조 내의 루트 노드에 연결될 적어도 하나의 새로운 리프 노드를 생성할 수 있다. 구체적으로, 프로세서(520)는 상기 생성된 적어도 하나의 새로운 리프 노드에 대해 원자성 쓰기를 수행할 수 있다.

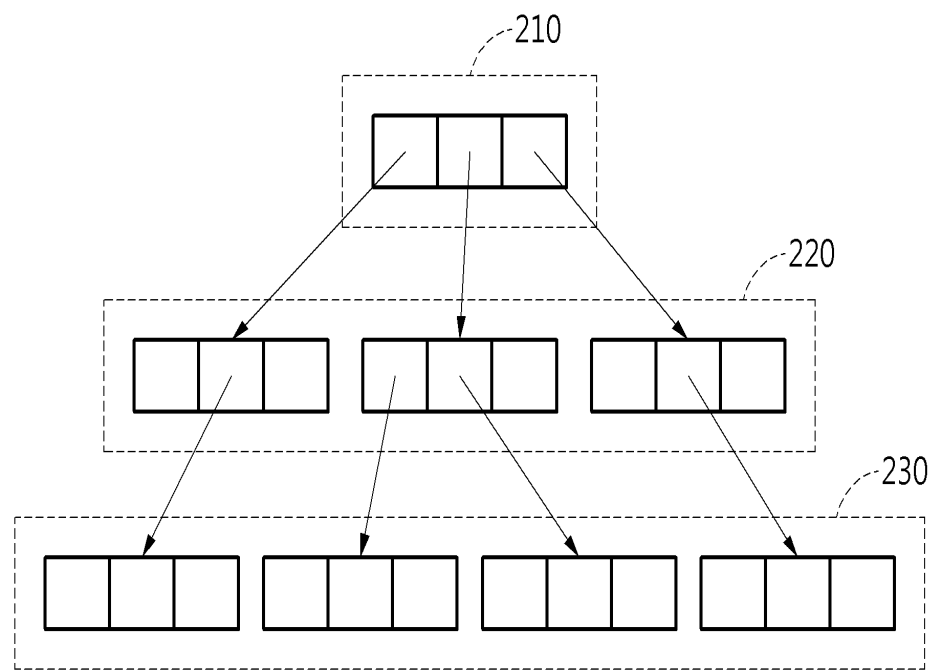
- [0052] 이어서, 프로세서(520)는 상기 적어도 하나의 새로운 리프 노드의 생성이 완료된 경우에, 상기 캐시 플러시 호출에 대응하여 래딕스 구조 내의 루트 노드에 원자성 쓰기를 수행하고, 상기 루트 노드와 상기 적어도 하나의 새로운 리프 노드를 연결할 수 있다.
- [0053] 도 6은 일실시예에 따른 비휘발성 메모리를 포함하는 데이터베이스 시스템의 관리 방법의 흐름도를 나타낸다.
- [0054] 비휘발성 메모리를 포함하는 데이터베이스 시스템의 관리 방법(600)은 노드의 엔트리에 포인터를 포함하는 래딕스 트리 구조로서 비휘발성 메모리에 데이터를 저장하는 단계(610) 및 프로세스 처리 과정에서 발생하는 캐시 플러시 호출에 대응하여 상기 래딕스 트리 내의 노드에 원자성 쓰기를 수행하는 단계(620)를 포함할 수 있다.
- [0055] 본 실시예에서 비휘발성 메모리는 하나의 컴퓨팅 시스템에 대응하는 저장 장치뿐만 아니라, 데이터를 중앙에서 제어할 수 있게 하여 데이터의 중복을 없애고 일관성을 유지하며, 데이터를 여러 사용자가 공동으로 사용하도록 지원하는 데이터베이스 시스템에 적용될 수 있다. 앞서 설명한 것과 같이, 위와 같은 비휘발성 메모리는 래딕스 구조의 데이터 구조로서 구현된 비휘발성 메모리를 나타낼 수 있다. 본 실시예에 따를 때, 종래의 파일 시스템(file system) 뿐만 아니라 데이터베이스 관리 시스템에도 래딕스 구조의 데이터 구조를 갖는 비휘발성 메모리가 이용될 수 있어 데이터의 일관성을 유지하는 원자성 쓰기가 수행되는 효과를 기대할 수 있다.
- [0056] 이상에서 설명된 실시예들은 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치, 방법 및 구성요소는, 예를 들어, 프로세서, 콘트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPGA(field programmable gate array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 콘트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.
- [0057] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치, 또는 전송되는 신호 파(signal wave)에 영구적으로, 또는 일시적으로 구체화(embodiment)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.
- [0058] 이상과 같이 실시예들이 비록 한정된 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기를 기초로 다양한 기술적 수정 및 변형을 적용할 수 있다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

도면

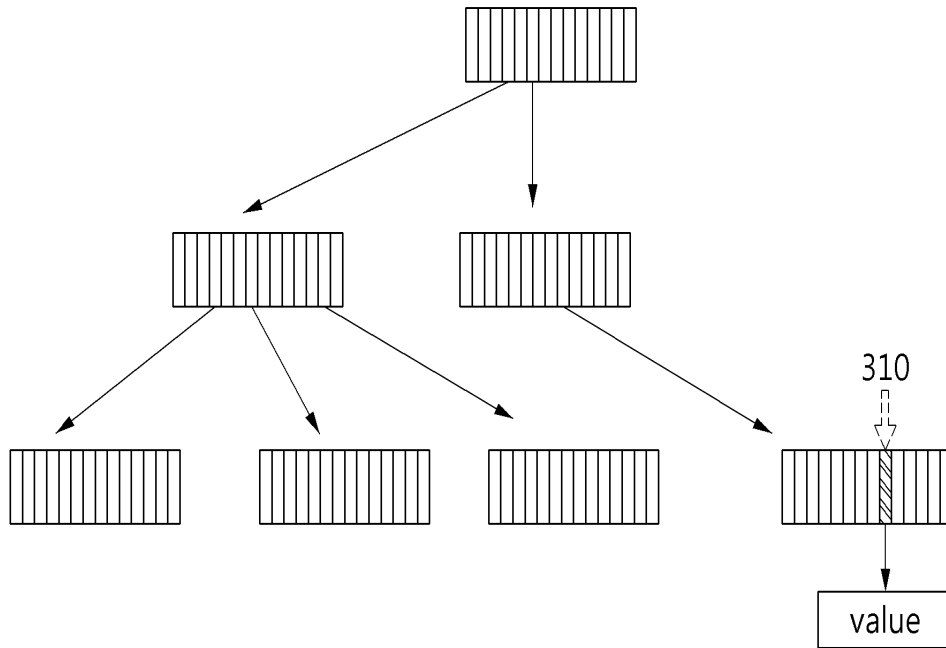
도면1



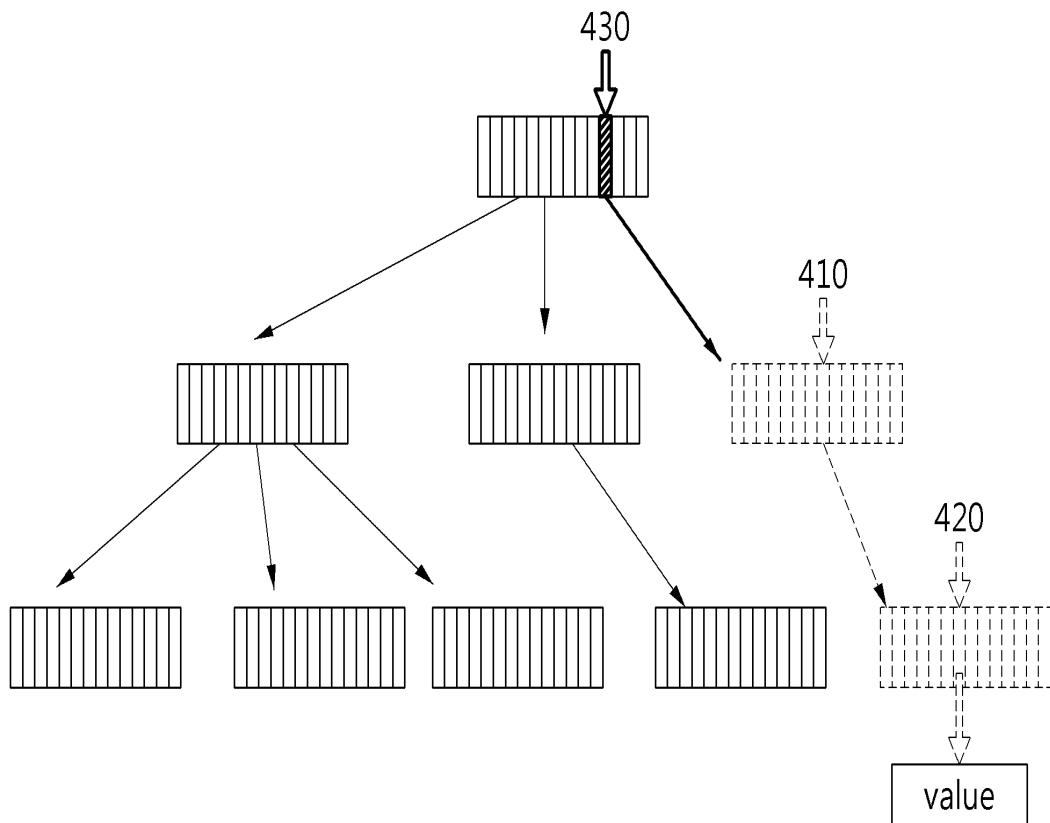
도면2



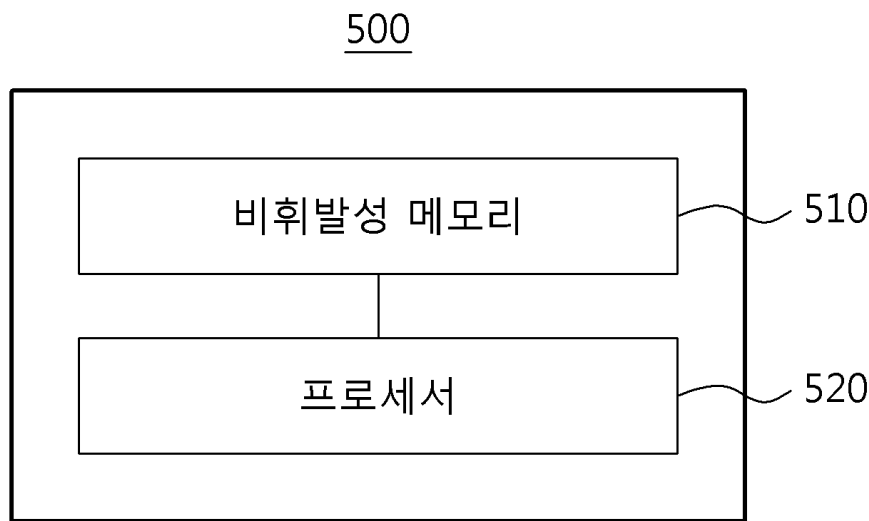
도면3



도면4



도면5



도면6

