(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2020-0015074
(43) 공개일자 2020년02월12일

(51) 국제특허분류(Int. Cl.)
G06F 12/0864 (2016.01) *G11C 16/26* (2006.01)
(52) CPC특허분류
G06F 12/0864 (2013.01)
G11C 16/26 (2013.01)
(21) 출원번호 10-2018-0090387
(22) 출원일자 2018년08월02일
심사청구일자 없음

(71) 출원인
에스케이하이닉스 주식회사
경기도 이천시 부발읍 경충대로 2091
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)
(72) 발명자
김선욱
경기도 남양주시 도농로 34, 214동 2003호(부영아파트1단지)
이원준
서울특별시 동대문구 무학로49길 14, 103호
한미선
대구광역시 달서구 구마로14남길 55, 409동 1202호(월성동, 주공아파트)
(74) 대리인
특허법인 이노

전체 청구항 수 : 총 40 항

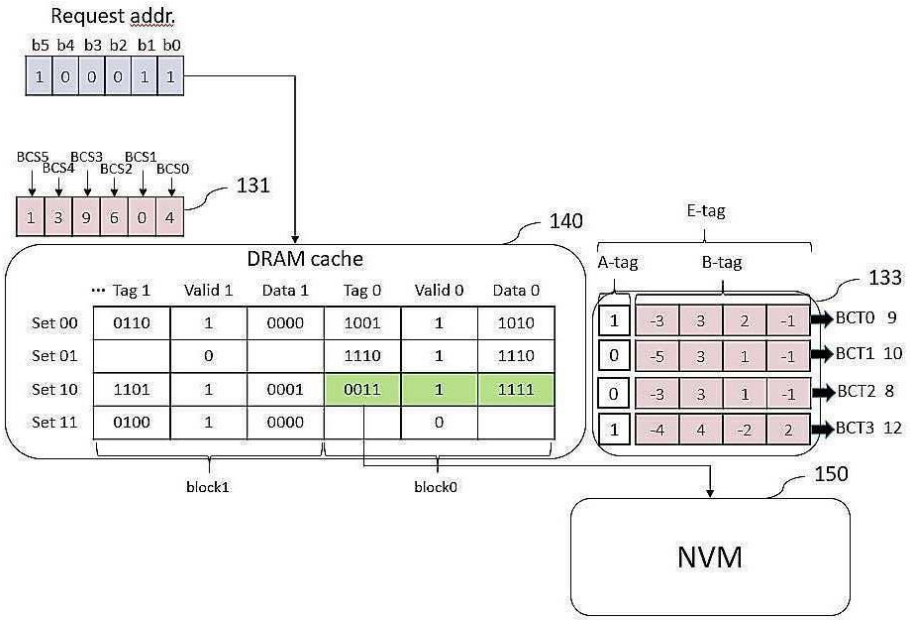
(54) 발명의 명칭 **비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템**

(57) 요 약

본 발명은 직전 주기에 적용된 직전 해시 함수와 현재 주기에 적용되는 현재 해시 함수를 선택적으로 이용하여 비휘발성 메모리로의 데이터 플러싱을 제한함으로써 시스템의 성능을 향상시킬 수 있는 비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템을 제공한다.

(뒷면에 계속)

대 표 도



본원의 제1 발명에 따른 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템은, 호스트 장치; 비휘발성 메모리; 상기 호스트 장치와 상기 비휘발성 메모리 간에 배치되고, 상기 호스트 장치의 데이터를 임시 저장하고, 복수의 세트를 포함하고, 상기 복수의 세트 각각은 복수의 블록을 포함하고, 상기 복수의 블록 각각은 기본 태그와 추가 태그가 결합된 확장 태그를 포함하는 DRAM cache; 상기 호스트 장치로부터 복수 비트의 리퀘스트 어드레스를 수신하는 메모리 콘트롤러; 및 상기 메모리 콘트롤러로부터 리퀘스트 어드레스를 수신하고, 복수의 세트용 비트 카운터를 이용하여 상기 리퀘스트 어드레스를 계수함으로써 상기 DRAM cache의 해시 함수를 결정하는 비트 카운터 콘트롤러를 포함한다.

(52) CPC특허분류
G06F 2212/305 (2013.01)

명 세 서

청구범위

청구항 1

호스트 장치;

비휘발성 메모리;

상기 호스트 장치와 상기 비휘발성 메모리 간에 배치되고, 상기 호스트 장치의 데이터를 임시 저장하고, 복수의 세트를 포함하고, 상기 복수의 세트 각각은 복수의 블록을 포함하고, 상기 복수의 블록 각각은 기본 태그와 추가 태그가 결합된 확장 태그를 포함하는 DRAM cache;

상기 호스트 장치로부터 복수 비트의 리퀘스트 어드레스를 수신하는 메모리 콘트롤러; 및

상기 메모리 콘트롤러로부터 리퀘스트 어드레스를 수신하고, 복수의 세트용 비트 카운터를 이용하여 상기 리퀘스트 어드레스를 계수함으로써 상기 DRAM cache의 해시 함수를 결정하는 비트 카운터 콘트롤러

를 포함하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 2

청구항 1에 있어서,

상기 비트 카운터 콘트롤러는 상기 DRAM cache 내 상기 복수의 블록 각각의 직전 해시 함수와 현재 해시 함수를 기억하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 3

청구항 2에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록에의 데이터 Read 요청 또는 데이터 Write 요청시, 상기 소정 블록의 직전 해시 함수나 현재 해시 함수로 hit 하면, 상기 비트 카운터 콘트롤러는 상기 블록의 데이터 상태를 'dirty'로 유지하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 4

청구항 2에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록의 데이터를 축출하고자 할 때, 상기 비트 카운터 콘트롤러는 상기 블록의 데이터를 플러싱 하고, 상기 소정 블록의 데이터 상태를 'dirty'에서 'invalid'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 5

청구항 2에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Read 요청시, 상기 소정 블록의 직전 해시 함수나 현재 해시 함수(Xc)로 hit 하면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'으로 유지하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 6

청구항 2에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Write 요청시, 상기 소정 블록의 직전 해시 함수나 현재 해시 함수로 hit 하면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'에서 'dirty'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 7

청구항 2에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록 내 데이터가 상기 비휘발성 메모리로 축출되면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'에서 'invalid'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 8

청구항 2에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'invalid'이고, 상기 소정 블록에의 데이터 Write 요청시, 상기 소정 블록의 직전 해시 함수와 현재 해시 함수가 모두 miss 이면, 상기 비트 카운터 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록의 데이터 상태를 'invalid'에서 'dirty'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 9

청구항 2에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'invalid'이고, 상기 소정 블록에의 데이터 Read 요청시, 상기 소정 블록의 직전 해시 함수와 현재 해시 함수(Xc)가 모두 miss 이면, 상기 비트 카운터 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록의 데이터 상태를 'invalid'에서 'clean'으로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 10

청구항 2에 있어서,

상기 직전 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록에 접근할 때 hit 하는 경우, 상기 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록에의 데이터 Write 요청이면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록을 상기 현재 해시 함수로 접근하여 데이터 Write를 수행하는 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 11

청구항 10에 있어서,

상기 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록에의 데이터 Read 요청이면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록 대신 상기 현재 해시 함수로 접근하는 블록에 데이터 Read를 수행하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 12

청구항 11에 있어서,

상기 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록의 데이터를 상기 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address로 축출하고자 할 때, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터를 상기 비휘발성 메모리의 address로 플러싱 하고, 상기 소정 블록의 데이터 상태를 'dirty'에서 'invalid'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 13

청구항 11에 있어서,

상기 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Write 요청이면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록을 상기 현재 해시 함수로 접근하는 블록에 데이터 Write를 수행하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 14

청구항 11에 있어서,

상기 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Read 요청이면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록 대신 상기 현재 해시 함수로 접근하는 블록에 데이터 Read를 수행하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 15

청구항 11에 있어서,

상기 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록 내 데이터가 상기 비휘발성 메모리로 축출되면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'에서 'invalid'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 16

청구항 11에 있어서,

상기 소정 DRAM cache 내 소정 블록이 invalid 상태에서, 상기 소정 블록에의 데이터 Write 요청시, 상기 소정 블록에 상기 직전 해시 함수와 현재 해시 함수로 접근하여 모두 miss 이면, 상기 비트 카운터 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록을 'invalid' 상태에서 'dirty' 상태로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 17

청구항 11에 있어서,

상기 DRAM cache 내 소정 블록이 invalid 상태에서, 상기 소정 블록에의 데이터 Read 요청시, 상기 소정 블록에 상기 직전 해시 함수와 현재 해시 함수로 접근하여 모두 miss 이면, 상기 비트 카운터 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록을 'invalid' 상태에서 'clean' 상태로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 18

청구항 2에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'dirty' 이고, 상기 소정 블록에의 데이터 Read 요청 또는 데이터 Write 요청시, 상기 현재 해시 함수로 hit 이면, 데이터 상태를 현 상태로 유지하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 19

청구항 18에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean' 이고, 상기 소정 블록에의 데이터 Write 요청시, 상기 현재 해시 함수로 hit 이면, 상기 데이터 상태를 'clean' 에서 'dirty'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 20

청구항 18에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean' 이고, 상기 소정 블록에의 데이터 Read 요청시, 상기 현재 해시 함수로 hit 이면, 상기 데이터 상태를 현 상태로 유지하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 21

호스트 장치;

비휘발성 메모리;

상기 호스트 장치와 상기 비휘발성 메모리 간에 배치되고, 상기 호스트 장치의 데이터를 임시 저장하고, 복수의 세트를 포함하고, 상기 복수의 세트 각각은 복수의 블록을 포함하고, 상기 복수의 블록 각각은 기본 태그와 추가 태그가 결합된 확장 태그를 포함하는 DRAM cache; 및

상기 호스트 장치로부터 복수 비트의 리퀘스트 어드레스를 수신하고, 복수의 세트용 비트 카운터를 이용하여 상기 리퀘스트 어드레스를 계수함으로써 상기 DRAM cache의 해시 함수를 결정하는 메모리 콘트롤러

를 포함하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 22

청구항 21에 있어서,

상기 메모리 콘트롤러는 상기 DRAM cache 내 상기 복수의 블록 각각의 직전 해시 함수와 현재 해시 함수를 기억하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 23

청구항 22에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록에의 데이터 Read 요청 또는 데이터 Write 요청시, 상기 소정 블록의 직전 해시 함수나 현재 해시 함수로 hit 하면, 상기 메모리 콘트롤러는 상기 블록의 데이터 상태를 'dirty'로 유지하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 24

청구항 22에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록의 데이터를 축출하고자 할 때, 상기 메모리 콘트롤러는 상기 블록의 데이터를 플러싱 하고, 상기 소정 블록의 데이터 상태를 'dirty'에서 'invalid'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 25

청구항 22에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Read 요청시, 상기 소정 블록의 직전 해시 함수나 현재 해시 함수(Xc)로 hit 하면, 상기 메모리 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'으로 유지하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 26

청구항 22에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Write 요청시, 상기 소정 블록의 직전 해시 함수나 현재 해시 함수로 hit 하면, 상기 메모리 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'에서 'dirty'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 27

청구항 22에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록 내 데이터가 상기 비휘발성 메모리로 축출되면, 상기 메모리 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'에서 'invalid'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 28

청구항 22에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'invalid'이고, 상기 소정 블록에의 데이터 Write 요청시, 상기 소정 블록의 직전 해시 함수와 현재 해시 함수가 모두 miss 이면, 상기 메모리 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데

이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록의 데이터 상태를 'invalid'에서 'dirty'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 29

청구항 22에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'invalid'이고, 상기 소정 블록에의 데이터 Read 요청시, 상기 소정 블록의 직전 해시 함수와 현재 해시 함수(Xc)가 모두 miss 이면, 상기 메모리 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록의 데이터 상태를 'invalid'에서 'clean'으로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 30

청구항 22에 있어서,

상기 직전 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록에 접근할 때 hit 하는 경우, 상기 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록에의 데이터 Write 요청이면, 상기 메모리 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록을 상기 현재 해시 함수로 접근하여 데이터 Write를 수행하는 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 31

청구항 30에 있어서,

상기 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록에의 데이터 Read 요청이면, 상기 메모리 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록 대신 상기 현재 해시 함수로 접근하는 블록에 데이터 Read를 수행하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 32

청구항 31에 있어서,

상기 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록의 데이터를 상기 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address로 축출하고자 할 때, 상기 메모리 콘트롤러는 상기 소정 블록의 데이터를 상기 비휘발성 메모리의 address로 플러싱 하고, 상기 소정 블록의 데이터 상태를 'dirty'에서 'invalid'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 33

청구항 31에 있어서,

상기 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Write 요청이면, 상기 메모리 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록을 상기 현재 해시 함수로 접근하는 블록에 데이터 Write를 수행하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 34

청구항 31에 있어서,

상기 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Read 요청이면, 상기 메모리 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록 대신 상기 현재 해시 함수로 접근하는 블록에 데이터 Read를 수행하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 35

청구항 31에 있어서,

상기 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록 내 데이터가 상기 비휘발성 메모리로 축출되면, 상기 메모리 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'에서 'invalid'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 36

청구항 31에 있어서,

상기 소정 DRAM cache 내 소정 블록이 invalid 상태에서, 상기 소정 블록에의 데이터 Write 요청시, 상기 소정 블록에 상기 직전 해시 함수와 현재 해시 함수로 접근하여 모두 miss 이면, 상기 메모리 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록을 'invalid' 상태에서 'dirty' 상태로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 37

청구항 31에 있어서,

상기 DRAM cache 내 소정 블록이 invalid 상태에서, 상기 소정 블록에의 데이터 Read 요청시, 상기 소정 블록에 상기 직전 해시 함수와 현재 해시 함수로 접근하여 모두 miss 이면, 상기 메모리 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록을 'invalid' 상태에서 'clean' 상태로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 38

청구항 22에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'dirty' 이고, 상기 소정 블록에의 데이터 Read 요청 또는 데이터 Write 요청시, 상기 현재 해시 함수로 hit 이면, 데이터 상태를 현 상태로 유지하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 39

청구항 38에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean' 이고, 상기 소정 블록에의 데이터 Write 요청시, 상기 현재 해시 함수로 hit 이면, 상기 데이터 상태를 'clean' 에서 'dirty'로 천이시키는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

청구항 40

청구항 38에 있어서,

상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean' 이고, 상기 소정 블록에의 데이터 Read 요청시, 상기 현재 해시 함수로 hit 이면, 상기 데이터 상태를 현 상태로 유지하는 것을 특징으로 하는 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템.

발명의 설명

기 술 분 야

[0001] 본 발명은 비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템에 관한 것으로, 더욱 상세하게는 DRAM을 cache로 사용하고 비휘발성 메모리를 저장 장치로 이용하는 하이브리드 메모리에서 DRAM cache에 비트 카운터 기법을 적용하여 비휘발성 메모리로의 플러시를 효율적으로 함으로써 시스템의 성능을 향상시킬 수 있는 비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템에 관한 것이다.

배 경 기 술

[0003] 일반적으로 비휘발성 메모리는 non-volatility, high scalability, byte-addressable 등의 장점으로 인하여 DRAM을 대체할 수 있는 메모리로 각광받고 있지만, DRAM에 비해 읽기/쓰기 성능이 현저히 낮고 수명이 짧다는 문제가 있다.

[0004] 비휘발성 메모리에 DRAM cache를 활용하는 하이브리드 메모리는 비휘발성 메모리로의 읽기/쓰기 요청 횟수를 줄여 메모리의 읽기/쓰기 성능을 향상시키며 비휘발성 메모리의 수명을 향상시킨다.

[0005] 하지만, DRAM cache 접근이 일부 set에 집중되어 발생하는 경우, 해당 set에 대한 cache miss가 증가하여 비휘발성 메모리로의 데이터 축출(eviction)이 증가하므로 읽기/쓰기 성능과 비휘발성 메모리의 수명을 감소시킨다.

[0006] 또한, 접근이 자주 발생하는 블록을 DRAM cache에서 비휘발성 메모리로 eviction을 하게 되면 비휘발성 메모리로의 쓰기 요청이 자주 발생하여 읽기/쓰기 성능과 비휘발성 메모리의 수명을 감소시킨다.

[0007] 이에 DRAM cache에 비트 카운터를 사용하여 index bits를 주기적으로 변경함으로써 DRAM cache 내 set를 고르게 사용할 필요가 있다. 이 경우, index bits를 주기적으로 변경할 때마다 DRAM cache 내 데이터를 비휘발성 메모리로 플러싱 하고, DRAM cache를 비워야 하므로, 시스템의 입장에서는 DRAM cache 내 set를 고르게 사용할 수 있다는 장점이 있는 반면, 주기적인 데이터 플러싱으로 인해 시스템에 부담으로 작용하게 되는 단점이 발생한다.

선행기술문헌

특허문헌

[0009] (특허문헌 0001) US 9092337 Apparatus, system, and method for managing eviction of data

(특허문헌 0002) US 9529730 Methods for cache line eviction

(특허문헌 0003) US 9886395 Evicting cached stores

발명의 내용

해결하려는 과제

[0010] 본 발명은 직전 주기에 적용된 직전 해시 함수와 현재 주기에 적용되는 현재 해시 함수를 선택적으로 이용하여

비휘발성 메모리로의 데이터 플러싱을 제한함으로써 시스템의 성능을 향상시킬 수 있는 비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템을 제공한다.

과제의 해결 수단

[0012] 본원의 제1 발명에 따른 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템은, 호스트 장치; 비휘발성 메모리; 상기 호스트 장치와 상기 비휘발성 메모리 간에 배치되고, 상기 호스트 장치의 데이터를 임시 저장하고, 복수의 세트를 포함하고, 상기 복수의 세트 각각은 복수의 블록을 포함하고, 상기 복수의 블록 각각은 기본 태그와 추가 태그가 결합된 확장 태그를 포함하는 DRAM cache; 상기 호스트 장치로부터 복수 비트의 리퀘스트 어드레스를 수신하는 메모리 콘트롤러; 및 상기 메모리 콘트롤러로부터 리퀘스트 어드레스를 수신하고, 복수의 세트용 비트 카운터를 이용하여 상기 리퀘스트 어드레스를 계수함으로써 상기 DRAM cache의 해시 함수를 결정하는 비트 카운터 콘트롤러를 포함한다.

[0013] 바람직하게는, 상기 비트 카운터 콘트롤러는 상기 DRAM cache 내 상기 복수의 블록 각각의 직전 해시 함수와 현재 해시 함수를 기억한다.

[0014] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록에의 데이터 Read 요청 또는 데이터 Write 요청시, 상기 소정 블록의 직전 해시 함수나 현재 해시 함수로 hit 하면, 상기 비트 카운터 콘트롤러는 상기 블록의 데이터 상태를 'dirty'로 유지한다.

[0015] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록의 데이터를 축출하고자 할 때, 상기 비트 카운터 콘트롤러는 상기 블록의 데이터를 플러싱 하고, 상기 소정 블록의 데이터 상태를 'dirty'에서 'invalid'로 천이시킨다.

[0016] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Read 요청시, 상기 소정 블록의 직전 해시 함수나 현재 해시 함수(Xc)로 hit 하면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'으로 유지한다.

[0017] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Write 요청시, 상기 소정 블록의 직전 해시 함수나 현재 해시 함수로 hit 하면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'에서 'dirty'로 천이시킨다.

[0018] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록 내 데이터가 상기 비휘발성 메모리로 축출되면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'에서 'invalid'로 천이시킨다.

[0019] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'invalid'이고, 상기 소정 블록에의 데이터 Write 요청시, 상기 소정 블록의 직전 해시 함수와 현재 해시 함수가 모두 miss 이면, 상기 비트 카운터 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록의 데이터 상태를 'invalid'에서 'dirty'로 천이시킨다.

[0020] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'invalid'이고, 상기 소정 블록에의 데이터 Read 요청시, 상기 소정 블록의 직전 해시 함수와 현재 해시 함수(Xc)가 모두 miss 이면, 상기 비트 카운터 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록의 데이터 상태를 'invalid'에서 'clean'으로 천이시킨다.

[0021] 바람직하게는, 상기 직전 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록에 접근할 때 hit 하는 경우, 상기 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록에의 데이터 Write 요청이면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록을 상기 현재 해시 함수로 접근하여 데이터 Write를 수행한다.

[0022] 바람직하게는, 상기 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록에의 데이터 Read 요청이면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록 대신 상기 현재 해시 함수로 접근하는 블록에 데이터 Read를 수행한다.

[0023] 바람직하게는, 상기 소정 블록의 데이터 상태가 'dirty'이고, 상기 소정 블록의 데이터를 상기 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address로 축출하고자 할 때, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터를 상기 비휘발성 메모리의 address로 플러싱 하고, 상기 소정 블록의 데이터 상태를 'dirty'에서 'invalid'로 천이시킨다.

[0024] 바람직하게는, 상기 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Write 요청이면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록을 상기 현재 해시 함수로 접근하는 블록에 데이터 Write를 수행한다.

[0025] 바람직하게는, 상기 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록에의 데이터 Read 요청이면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'invalid'로 천이시키고, 상기 소정 블록 대신 상기 현재 해시 함수로 접근하는 블록에 데이터 Read를 수행한다.

[0026] 바람직하게는, 상기 소정 블록의 데이터 상태가 'clean'이고, 상기 소정 블록 내 데이터가 상기 비휘발성 메모리로 축출되면, 상기 비트 카운터 콘트롤러는 상기 소정 블록의 데이터 상태를 'clean'에서 'invalid'로 천이시킨다.

[0027] 바람직하게는, 상기 소정 DRAM cache 내 소정 블록이 invalid 상태에서, 상기 소정 블록에의 데이터 Write 요청시, 상기 소정 블록에 상기 직전 해시 함수와 현재 해시 함수로 접근하여 모두 miss 이면, 상기 비트 카운터 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록을 'invalid' 상태에서 'dirty' 상태로 천이시킨다.

[0028] 바람직하게는, 상기 DRAM cache 내 소정 블록이 invalid 상태에서, 상기 소정 블록에의 데이터 Read 요청시, 상기 소정 블록에 상기 직전 해시 함수와 현재 해시 함수로 접근하여 모두 miss 이면, 상기 비트 카운터 콘트롤러는 상기 현재 해시 함수를 이용하여 상기 DRAM cache 내 소정 블록의 address에 대응하는 상기 비휘발성 메모리의 address에 저장된 데이터를 상기 DRAM cache 내 소정 블록의 address로 읽어들이고, 상기 DRAM cache 내 소정 블록을 'invalid' 상태에서 'clean' 상태로 천이시킨다.

[0029] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'dirty' 이고, 상기 소정 블록에의 데이터 Read 요청 또는 데이터 Write 요청시, 상기 현재 해시 함수로 hit 이면, 데이터 상태를 현 상태로 유지한다.

[0030] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean' 이고, 상기 소정 블록에의 데이터 Write 요청시, 상기 현재 해시 함수로 hit 이면, 상기 데이터 상태를 'clean' 에서 'dirty'로 천이시킨다.

[0031] 바람직하게는, 상기 DRAM cache 내 소정 블록의 데이터 상태가 'clean' 이고, 상기 소정 블록에의 데이터 Read 요청시, 상기 현재 해시 함수로 hit 이면, 상기 데이터 상태를 현 상태로 유지한다.

[0032] 또한, 본원의 제2 발명에 따른 비휘발성 메모리 모듈을 이용한 컴퓨팅 시스템은, 호스트 장치; 비휘발성 메모리; 상기 호스트 장치와 상기 비휘발성 메모리 간에 배치되고, 상기 호스트 장치의 데이터를 임시 저장하고, 복수의 세트를 포함하고, 상기 복수의 세트 각각은 복수의 블록을 포함하고, 상기 복수의 블록 각각은 기본 태그와 추가 태그가 결합된 확장 태그를 포함하는 DRAM cache; 및 상기 호스트 장치로부터 복수 비트의 리퀘스트 어드레스를 수신하고, 복수의 세트용 비트 카운터를 이용하여 상기 리퀘스트 어드레스를 계수함으로써 상기 DRAM cache의 해시 함수를 결정하는 메모리 콘트롤러를 포함한다.

발명의 효과

[0034] 본 발명에 따르면, 직전 주기에 적용된 직전 해시 함수와 현재 주기에 적용되는 현재 해시 함수를 선택적으로 이용하여 비휘발성 메모리로의 데이터 플러싱을 제한함으로써 시스템의 성능을 향상시킬 수 있다.

도면의 간단한 설명

[0036] 도 1은 본 발명의 일실시예에 따른 비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템의 전체 블록도,

도 2는 본 발명의 다른 실시예에 따른 비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템의 전체 블록도,

도 3은 본 발명의 일실시예에 따른 비휘발성 메모리 모듈 내 비트 카운트를 설명하는 도면,

도 4는 본 발명의 일실시예에 따른 cache DRAM 내 address bits에 대한 access request 처리시 비트 카운터 값 변화를 설명하기 위한 도면,

도 5는 본 발명의 일실시예에 따라 cache DRAM에 BCS 기법을 적용하기 전과 후의 request miss 분포를 나타내는 도면,

도 6은 본 발명의 일실시예에 따라 BCS에서 해시 함수의 재설정을 설명하는 도면,

도 7은 본 발명의 일실시예에 따라 해시 함수의 재설정시 일부 캐시 블록만 플러싱 하는 기법의 예시도,

도 8은 본 발명의 일실시예에 따른 캐시 블록의 상태 천이 프로토콜 예시도, 및

도 9는 본 발명의 다른 실시예에 따른 캐시 블록의 상태 천이 프로토콜 예시도이다.

발명을 실시하기 위한 구체적인 내용

[0037] 본 발명의 이점 및 특징, 그리고 그것들을 달성하는 방법은 첨부되는 도면과 함께 상세하게 후술되어 있는 실시예를 참조하면 명확해질 것이다. 그러나 본 발명은 이하에서 개시되는 실시 예에 한정되는 것이 아니라 서로 다른 다양한 형태로 구현될 수 있으며, 단지 본 실시 예는 본 발명의 개시가 완전하도록 하고, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 발명의 범주를 완전하게 알려주기 위해 제공되는 것이며, 본 발명은 청구항의 범주에 의해 정의될 뿐이다.

[0038] 본 명세서에서 사용된 용어는 실시 예들을 설명하기 위한 것이며 본 발명을 제한하고자 하는 것은 아니다. 본 명세서에서, 단수형은 문구에서 특별히 언급하지 않는 한 복수형도 포함한다. 명세서에서 사용되는 '포함한다(comprises)' 및/또는 '포함하는(comprising)'은 언급된 구성요소, 단계, 동작 및/또는 소자는 하나 이상의 다른 구성요소, 단계, 동작 및/또는 소자의 존재 또는 추가를 배제하지 않는다.

[0039] 하나의 소자(elements)가 다른 소자와 '접속된(connected to)' 또는 '커플링된(coupled to)' 이라고 지칭되는 것은, 다른 소자와 직접 연결 또는 커플링 된 경우 또는 중간에 다른 소자를 개재한 경우를 모두 포함한다. 반면, 하나의 소자가 다른 소자와 '직접 접속된(directly connected to)' 또는 '직접 커플링된(directly coupled to)'으로 지칭되는 것은 중간에 다른 소자를 개재하지 않은 것을 나타낸다. '및/또는'은 언급된 아이템들의 각각 및 하나 이상의 모든 조합을 포함한다.

[0040] 명세서 전문에 걸쳐 동일한 참조 부호는 동일한 구성 요소를 지칭한다. 따라서, 동일한 참조 부호 또는 유사한 참조 부호들은 해당 도면에서 언급 또는 설명되지 않았더라도, 다른 도면을 참조하여 설명될 수 있다. 또한, 참조 부호가 표시되지 않았더라도, 다른 도면들을 참조하여 설명될 수 있다.

[0042] 본 발명에서 캐시 축출(cache eviction)이라 함은 DRAM cache의 특정 set에서 더 이상 저장할 공간이 없을 경우, 기존에 저장된 block의 데이터를 비휘발성 메모리로 축출하는 행위를 의미한다.

[0043] 또한, 본 발명에서 플러싱(flush, flushing)이라 함은 DRAM cache의 데이터를 비휘발성 메모리에 덮어 쓰게 하고, DRAM cache를 초기화하는 행위를 의미한다.

[0044] 또한, 본 발명에 따르면, DRAM cache는 식별 가능한 복수의 세트를 포함하고, 개별 세트는 식별 가능한 복수의 블록을 포함하며, 개별 블록은 서로 식별 가능하도록 각각의 tag를 포함한다.

[0046] 도 1은 본 발명의 일실시예에 따른 비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템의 전체 블록도이다.

[0047] 본 발명의 일실시예에 따른 비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템은 호스트 장치(110), 메모리 콘트롤러(120), 비트 카운터 콘트롤러(130), DRAM cache(140), 및 비휘발성 메모리(150)를 포함한다.

[0048] 호스트 장치(110)는 호스트, 서버, 저장 영역 네트워크(SAN: storage area network)의 저장 콘트롤러, 워크스테이션, 개인용 컴퓨터, 랩터 컴퓨터, 핸드헬드 컴퓨터, 수퍼컴퓨터, 컴퓨터 클러스터, 네트워크 스위치, 라우터, 또는 응용기기, 데이터베이스 또는 저장 응용기기, 데이터 획득 또는 데이터 캡처 시스템, 진단 시스템, 테스트 시스템, 로봇, 포터블 전자 기기, 무선 기기 등일 수 있다.

[0049] 본 발명의 일실시예에 따르면, 호스트 장치(110)는 메모리 콘트롤러(120)와 통신할 수 있다. 메모리 콘트롤러(120)는 호스트 장치(110)로부터 read/write request와 request address를 수신한다.

[0050] 본 발명의 일실시예에 따르면, 메모리 콘트롤러(120)는 비트 카운터 콘트롤러(130)와 통신할 수 있다. 비트 카운터 콘트롤러(130)는 메모리 콘트롤러(120)로부터 read/write request와 request address를 수신하고, 메모리 콘트롤러(120)로부터 수신되는 request address를 복수의 세트용 비트 카운터(131, BCS: Bit Counter for Set)와 복수의 태그용 비트 카운터(133, BCT: Bit Counter for Tag)를 이용하여 변경된 request address를 생성하여 출력하고, 데이터를 DRAM cache에 캐싱하고, 캐싱된 데이터를 DRAM cache(140)에서 비휘발성 메모리(150)로 축출(evict) 할 수 있다. 여기서, DRAM cache(140)에서 비휘발성 메모리(150)로 축출한다는 것은 DRAM cache(140)의 소정 address에 저장된 데이터를 상기 DRAM cache(140)의 소정 address와 동일한 address를 가지는 비휘발성 메모리(150) 내 대응 address에 저장하고, 상기 DRAM cache(140)의 소정 address에 저장된 데이터를 삭제하는 것을 의미한다. 한편, 일반적으로 DRAM cache의 저장 능력은 비휘발성 메모리의 저장 능력에 비해 현저히 작고, 비휘발성 메모리의 일부 영역에는 DRAM cache 내 다수의 address와 동일한 address들이 존재한다.

[0051] 본 발명에 따르면, 비휘발성 메모리(150, NVM: non-volatile memory)는 예컨대, ReRAM(resistive RAM)인 경우, 가변 저항부는 저항체(Resistive Material, PCMO막 등)으로 형성될 수 있고, PCRAM(phase change RAM)인 경우 칼코게나이드막으로 형성될 수 있으며, MRAM인 경우 자화물질의 자성층으로 형성될 수 있고, STTRAM(spin-transfer torque RAM)인 경우, 자화물질의 자화 반전 소자층으로 형성될 수 있다.

[0052] 본 발명의 일실시예에 따르면, 비트 카운터 콘트롤러(130), DRAM cache(140) 및 비휘발성 메모리(150)가 하나의 모듈 형태로 구현될 수 있다.

[0054] 도 2는 본 발명의 다른 실시예에 따른 비휘발성 메모리 모듈을 포함하는 컴퓨팅 시스템의 전체 블록도이다.

[0055] 본 발명의 다른 실시예에 따르면, 메모리 콘트롤러(220) 내에 복수의 세트용 비트 카운터(221)와 복수의 태그용 비트 카운터(223)가 구현될 수 있다.

[0056] 본 발명의 일실시예에 따르면, 호스트 장치(110)는 메모리 콘트롤러(220)와 통신할 수 있다. 메모리 콘트롤러(120)는 호스트 장치(110)로부터 read/write request와 request address를 수신하고, 호스트 장치(110)로부터 수신되는 request address를 복수의 세트용 비트 카운터(221)와 복수의 태그용 비트 카운터(223)를 이용하여 변경된 request address를 생성하여 출력하고, 데이터를 DRAM cache(140)에 캐싱하고, 캐시 처리된 데이터를 DRAM cache(140)에서 비휘발성 메모리(150)로 축출할 수 있다.

[0057] 본 발명의 일실시예에 따르면, DRAM cache(140) 및 비휘발성 메모리(150)가 하나의 모듈 형태로 구현될 수 있다.

[0059] 도 3은 본 발명의 일실시예에 따른 비휘발성 메모리 모듈 내 비트 카운터 동작을 설명하는 도면으로서, 도 1에 기반하여 설명하기로 한다.

[0060] 도 3에 도시된 바와 같이, DRAM cache(140)는 4개의 sets(set 00, set 01, set 10, set 11)를 포함하고, 개별 set는 복수의 블록(block0 ~ 3)를 포함하고, 개별 블록은 각각의 태그(Tag 0, Tag 1, ...)를 포함하고, 개별 태그마다 소정 크기의 데이터(Data0, Data1, ...)를 저장할 수 있다. 도 3에서는 제3 블록 및 제4 블록(block 2, block3)을 생략하였다.

[0061] 비트 카운터 콘트롤러(130)가 메모리 콘트롤러(120)로부터 read/write request와 request address(b5, b4, b3, b2, b1, b0)를 수신하면, 비트 카운터 콘트롤러(130) 내 6개의 세트용 비트 카운터(131: BCS0, BCS1, BCS2, BCS3, BCS4, BCS5)는 각각 순차적으로 수신되는 request address를 소정 횟수 동안 계수한다. 여기서 6 bits의 request address 중 임의의 2 bits를 index로 사용하고, 나머지 4 bits를 B-tag로 사용할 수 있다.

[0062] 한편, 본 발명에 따르면, DRAM cache 내 모든 블록은 B-tag(Basic tag)에 A-tag(Added tag)가 결합된 E-tag(Extended tag)를 구비한다. 여기서, A-tag는 현재 구간에 생성된 해시 함수로 해당 블록에 접근하는지 혹은 직전 구간에 생성된 해시 함수로 해당 블록에 접근하는지를 표시한다. 즉, 본 발명에 따르면, A-tag로는 구간마다 "0"과 "1"을 교번적으로 사용한다.

[0063] 예컨대, 수신되는 request address가 6 bits이므로, 6개의 세트용 비트 카운터(131)가 필요하고, 세트용 비트

카운터가 2^{10} 횟수 동안 계수한다면, 개별 세트용 비트 카운터는 10개의 비트 카운터로 구현될 수 있다. 여기서 10개의 비트 카운터 내 중 1개는 부호 표시용이고, 나머지 9개는 비트 카운트용이다.

[0064] 그리고, 비트 카운터 콘트롤러(130)는 개별 세트용 비트 카운터(131)가 계수한 6비트에 대하여 각각의 절대치를 생성하고, 생성된 절대치를 2개씩 묶어 15개의 조합을 생성하여 합산한다. 즉, 6개 중 2개를 선택하는 조합은 6C2=15에 의해 만들어질 수 있다. 예로써, (b5, b4), (b5, b3), (b5, b2), (b5, b1), (b5, b0), (b4, b3), (b4, b2), (b4, b1), (b4, b0), (b3, b2), (b3, b1), (b3, b0), (b2, b1), (b2, b0), (b1, b0) 등일 수 있다.

[0065] 계속해서, 비트 카운터 콘트롤러(130)는 15개의 조합에 대하여 합산한 절대치 합산값의 크기를 비교하고, 상대적으로 가장 작은 절대치 합산값을 가진 조합을 DRAM cache의 index용 해시 함수로 선택할 수 있다. 여기서, 해시 함수(hash function)라 함은 비트 카운터 콘트롤러(130)가 호스트 장치로부터 수신되는 read/write request와 request address를 이용하여 index bits를 찾아내기 위한 인자(factor)를 의미한다. 한편, 절대치를 생성하는 이유는 도 4에서 설명하기로 한다.

[0066] 이와 같이 소정 횟수 동안 request address를 계수하고, 계수한 절대치 합산값이 가장 작은 조합을 해시 함수로 선택하게 되면, 소정 횟수 동안 해당 set로의 접근 횟수가 나머지 set로의 접근 횟수에 비해 상대적으로 가장 빈번하였음을 의미한다. 이에 따라 DRAM cache 내 set를 고르게 사용할 수 있는 효과가 있다. 이에 대해서는 도 5에서 예를 들어 설명하기로 한다.

[0068] 비트 카운터 콘트롤러(130) 내 태그용 비트 카운터(133)는, request miss가 발생할 때마다 해당 set 내의 개별 tag bits를 계수하고, tag bits의 계수한 절대치 합산값이 가장 작은 블록을 선택하여 비휘발성 메모리(150) 내 DRAM cache의 address와 동일한 address로 축출한다. 본 발명에 따르면, 태그용 비트 카운터(133)는 세트별로 하나씩 구현될 수 있고, 4개의 sets(set 00, set 01, set 10, set 11)를 포함하므로, 비트 카운터 콘트롤러(130)는 4개의 태그용 비트 카운터를 포함하여 구현될 수 있다.

[0069] 비트 카운터(131, 133)의 동작은 도 4 및 도 5에서 상세하게 설명하기로 한다.

[0071] 도 4는 본 발명의 일실시예에 따른 DRAM cache 내 address에 대한 access request 처리시 세트용 비트 카운터 값 변화를 설명하기 위한 도면이다.

[0072] 본 발명에 따르면, 각각의 request address를 계수하는 방법은 request address가 "1"이면 (+)1로 계수하고, request address가 "0"이면 (-)1로 계수한다.

[0073] 초기에는 제6 내지 제1 세트용 비트 카운터(BCS5 ~ BCS0)에 "0 0 0 0 0 0" 값이 저장되어 있다가, 제1 address(Addr0) "1 1 0 1 1 0"이 입력되면, 위와 같은 본 발명에 따른 request address 계수 방법에 따라 제6 내지 제1 세트용 비트 카운터(BCS5 ~ BCS0)는 "1 1 -1 1 1 -1"로 계수한다.

[0074] 이후 제2 address(Addr1) "1 0 0 1 0 1"이 입력되면, 제6 내지 제1 세트용 비트 카운터(BCS5 ~ BCS0)는 "2 0 -2 2 0 0" 로 계수한다.

[0075] 이러한 방식으로, 제m address(Addr(m-1)) "1 1 0 0 0 1"이 입력되면, 제6 내지 제1 세트용 비트 카운터(BCS5 ~ BCS0)는 "-5 5 9 6 0 4"로 계수한다.

[0076] 그런데 6개의 세트용 비트 카운터에 계수된 15개의 비트 조합에 대하여 절대치를 취하고, 이들을 가산한다. 이들 가운데 비트 조합(b1, b0)에 대응하는 제1 및 제2 세트용 비트 카운터(BCS0, BCS1) 각각에 계수된 값 "4" "0"에 절대치를 취하고, 이들을 가산한 값(4+0=4)이 가장 작다. 따라서 비트 조합(b1, b0)을 DRAM cache의 해시 함수로 선택한다.

[0077] 그리고 6 bits의 request address 중 나머지 4 bits(b5 ~ b2)를 tag로 사용한다.

[0078] 이와 같이 비트 카운터 콘트롤러(130)는 소정 횟수 동안 입력되는 request address를 계수하여 절대치 합산값이 가장 작은 비트 조합을 해시 함수로 선택함으로써 DRAM cache 내 set를 고르게 사용할 수 있는 효과가 있다.

[0080] 도 5a는 DRAM cache에 BCS 기법을 적용하기 전의 request miss 분포를 나타내는 도면이고, 도 5b는 본 발명의

일실시예에 따라 DRAM cache에 BCS 기법을 적용한 후의 cache miss 분포를 나타내는 도면이다.

[0081] 예컨대, 도 5a에 도시된 DRAM cache에 BCS 기법을 적용하기 전의 request miss 분포를 보면, index bits를, 하위 비트(b1, b0)로 고정하여 access request를 처리하는 경우, 특정 set로의 접근이 집중되는 것을 볼 수 있다. 즉, 전체 6회의 접근 중 5회를 index bits "0 0"에 접근하게 된다. 이로 인해 4회의 request miss가 발생하여 비휘발성 메모리에 4회의 write requests가 발생한다.

[0082] 하지만, 도 5b에 도시된 바와 같이, 본 발명의 일실시예에 따라 DRAM cache에 BCS 기법을 적용하면, access requests로 동일한 request address가 입력되는 경우에도, set별 접근 집중도를 완화시킬 수 있다.

[0083] 예컨대, 비트 조합(b5, b4)에 대응하는 제5 및 제6 세트용 비트 카운터(BCS4, BCS5)에 각각 계수된 값 "0" "0"에 절대치를 취하고, 이들을 가산하면 절대치 합산값은 0이다. 비트 조합(b3, b2)에 대응하는 제3 및 제4 세트용 비트 카운터(BCS2, BCS3)에 각각 계수된 값 "2" "2"에 절대치를 취하고, 이들을 가산하면 절대치 합산값은 4이다. 비트 조합(b1, b0)에 대응하는 제1 및 제2 세트용 비트 카운터(BCS0, BCS1)에 각각 계수된 값 "0" "1"에 절대치를 취하고, 이들을 가산하면 절대치 합산값은 1이다.

[0084] 여기서, 15개의 조합에 대하여 모두 절대치를 취하고 가산하여 절대치가 가장 작은 조합을 선별해야 하나, 설명의 편의를 위하여 3개의 조합에 대해서만 설명하였음을 이해하여야 한다.

[0085] 따라서 비트 조합(b5, b4)을 해시 함수로 선택한다. 이 경우, access requests가 set_index bits "0 0"로 1회, set_index bits "0 1"로 2회, set_index bits "1 0"으로 2회, set_index bits "1 1"로 1회가 각각 발생하여 세트별 접근이 분산됨을 알 수 있다. 이로 인해 2회의 request miss가 발생하므로 비휘발성 메모리로의 write request도 2회로 줄어드는 것을 알 수 있다.

[0087] 도 6은 본 발명의 일실시예에 따라 BCS에서 해시 함수의 재설정을 설명하는 도면이다.

[0088] 본 발명에 따르면, 비트 카운터 콘트롤러(130)는 6개의 세트용 비트 카운터(131)를 이용하여 해시 함수를 소정 횟수마다 재설정하고, 특정 address를 새로운 해시 함수로 결정하는 순간 DRAM cache(140) 내 데이터들을 비휘발성 메모리(150)로 플러싱(flush) 하여 DRAM cache(140)를 삭제한다.

[0089] 즉, 비트 카운터 콘트롤러(130)는 이전 구간 Previous Interval(i-1) 동안 세트용 비트 카운터에 카운트 값이 누적되면 현재 구간 Current Interval(i)에서 사용할 해시 함수를 재설정하고, DRAM cache(140) 내 데이터를 비휘발성 메모리(150)로 플러시(flush) 한다.

[0090] 이후 동일한 방식으로 비트 카운터 콘트롤러(130)는 현재 구간 Current Interval(i) 동안 세트용 비트 카운터에 카운트 값이 누적되면 다음 구간 Next Interval(i+1)에서 사용할 해시 함수를 재설정하고, DRAM cache(140) 내 데이터를 비휘발성 메모리(150)로 플러시(flush) 한다.

[0091] 여기서, 소정 횟수마다 DRAM cache의 데이터를 비휘발성 메모리로 플러시 하는 이유는 다음과 같다.

[0092] 이전 구간 Previous Interval(i-1)에 적용되는 해시 함수와 현재 구간 Current Interval(i)에 적용되는 해수 함수가 다르거나 현재 구간 Current Interval(i)에 적용되는 해수 함수와 다음 구간 Next Interval(i+1)에 적용되는 해수 함수가 다르게 되면 소정 데이터를 저장하는 DRAM cache의 address와 비휘발성 메모리의 address가 불일치하게 된다.

[0093] 이러한 문제점을 해소하기 위하여 개별 구간마다 DRAM cache의 address와 비휘발성 메모리의 address를 일치시키기 위해 매 구간이 시작될 때마다 DRAM cache(140) 내 데이터를 비휘발성 메모리(150)로 플러시(flush) 할 필요가 있다.

[0095] 도 7은 본 발명의 일실시예에 따라 해시 함수의 재설정시 일부 캐시 블록만 플러시 하는 기법의 예시도이다.

[0096] 본 발명의 일실시예에 따르면, 비트 카운터 콘트롤러(130)는 6 bits의 request address 중 2 bits의 index를 제외한 나머지 4 bits를 개별 블록의 B-tag로 사용한다.

[0097] 도 7에 도시된 바와 같이, DRAM cache(140) 내 i번째 set에 4개의 블록(block0 ~ block3)을 위한 각각의 B-tag(tag0 ~ tag3)가 이미 저장되어 있다고 가정하자. 이때 DRAM cache(140) 내 제2 블록(blokc1)에 새로운 B-tag를 저장하고자 하면, DRAM cache(140) 내 제2 블록(block1)에 request address용 B-tag가 이미 저장되어 있

기 때문에 request miss가 발생한다.

[0098] request miss가 발생하면, 비트 카운터 콘트롤러(130)는 해당 세트의 태그용 비트 카운터(133)에 저장된 값과 개별 블록(block0 ~ block3)의 B-tag(tag0 ~ tag3)를 각각 가산한다. 예컨대, 태그용 비트 카운터(133)에 저장된 값(-4 2 2 0)과 제4 B-tag(tag3)의 태그 비트(1 1 0 0)를 가산하면 (-3 3 2 -1)이 된다. 태그용 비트 카운터(133)에 저장된 값(-4 2 2 0)과 제3 B-tag(tag2)의 태그비트(0 1 0 0)를 가산하면 (-5 3 1 -1)이 된다. 태그용 비트 카운터(133)에 저장된 값(-4 2 2 0)과 제2 B-tag(tag1)의 태그 비트(1 1 0 1)를 가산하면 (-3 3 1 -1)이 된다. 마지막으로, 태그용 비트 카운터(133)에 저장된 값(-4 2 2 0)과 제1 B-tag(tag0)의 태그 비트(0 1 0 1)를 가산하면 (-4 4 -2 2)이 된다.

[0099] 본 발명에 따르면, 해당 세트의 태그용 비트 카운터(133)에 저장된 값과 개별 블록(block0 ~ block3)의 B-tag(tag0 ~ tag3)를 각각 가산하는 방법은 request address가 "1"이면 (+)1로 계수하고, request address가 "0"이면 (-)1로 계수한다.

[0100] 비트 카운터 콘트롤러(130)는 가산한 축출 후보 블록별 결과값의 절대치 합산값이 가장 작은 블록을 축출 블록으로 선택한다. 예컨대, 제2 블록(block1)의 절대치 합산값이 8로 가장 작기 때문에 제2 블록(block1)을 축출 블록으로 선택한다.

[0101] 비트 카운터 콘트롤러(130)는 DRAM cache(140) 내 선택된 축출 블록인 제2 블록(block1)을 비휘발성 메모리(150)로 축출한다.

[0102] 비트 카운터 콘트롤러(130)는 호스트 장치(110)로부터 입력되는 새로운 request address를 DRAM cache(140) 내 비어 있는 블록인 제2 블록(block1)에 삽입한다.

[0103] 마지막으로, 비트 카운터 콘트롤러(130)는 축출되는 tag(1 1 0 1)와 기 저장된 태그용 비트 카운터 결과값 (-4 2 2 0)에 가산하여 새로운 태그용 비트 카운터 결과값 (-3 3 1 -1)으로 갱신한다.

[0105] 본 발명에 따르면, A-tag로는 구간마다 "0"과 "1"을 교번적으로 사용하는바, 이전 구간(i-1)의 A-tag가 "0"이면 해당 블록에 접근한 것으로 세트 된 상태에서, 이전 구간(i-1)의 A-tag가 "1"이라면 이전 구간(i-1)에서는 해당 블록에 접근하지 않은 것을 의미한다. 그리고, 이전 구간(i-1)의 A-tag가 "1"인 경우, 해당 블록에 접근이 이루어지면 비트 카운터 콘트롤러(130)는 해당 블록의 A-tag를 "1"에서 "0"으로 전환시킨다.

[0106] 예컨대, 이전 구간(i-1)에서 생성된 해시 함수로 해당 블록에 접근하면 A-tag를 "0"으로 세트한다. 이전 구간(i-1)에서 현재 구간(i)으로 전환하는 순간, A-tag를 확인하여 이전 구간(i-1)에서 생성된 해시 함수로 접근된 블록은 DRAM cache 내에 데이터를 유지하는 한편, 그 전 구간(i-2)에서 생성된 해시 함수로 접근된 블록이면 DRAM cache 내 데이터를 비휘발성 메모리로 플러시 한다.

[0107] 즉, 이전 구간(i-1)에서 현재 구간(i)으로 전환하는 순간, A-tag가 "0"이면 DRAM cache 내에 데이터를 유지하는 한편, A-tag가 "1"이면 DRAM cache 내 데이터를 비휘발성 메모리로 플러시 한다. 다시 말해서, 이전 구간(i-1)에서 현재 구간(i)으로 전환하는 순간, 임의 블록의 A-tag가 "0"이 아니고 "1"이면, 해당 블록은 이전 구간(i-1) 동안 접근되지 않았음을 의미한다.

[0108] 이후, 현재 구간(i)에서 생성된 해시 함수로 소정 블록에 접근하면 해당 블록의 A-tag를 "1"로 세트한다. 현재 구간(i)에서 다음 구간(i+1)으로 전환하는 순간, A-tag를 확인하여 해당 구간(i)에서 생성된 해시 함수로 접근한 블록은 DRAM cache 내에 데이터를 유지하는 한편, 이전 구간(i-1)에서 생성된 해시 함수로 접근한 블록이면 DRAM cache 내 데이터를 비휘발성 메모리로 플러시 한다.

[0109] 즉, 현재 구간(i)에서 다음 구간(i+1)으로 전환하는 순간, 소정 블록의 A-tag가 "1"이면 DRAM cache 내 해당 블록의 데이터를 유지하는 한편, 소정 블록의 A-tag가 "0"이면 DRAM cache 내 데이터를 비휘발성 메모리로 플러시 한다.

[0110] 결국, DRAM cache 내에는 이전 구간에 접근이 발생한 블록과, 현재 구간에 접근이 발생한 블록으로 구분할 수 있다. 본 발명에 따르면, DRAM cache 내 블록의 접근을 위하여 이전 구간의 해시 함수를 저장하기 위한 영역이 마련되고, 매 구간마다 이전 구간의 해시 함수를 갱신한다.

[0111] 이를 통해 본 발명에서는 구간마다 DRAM cache 내 모든 블록을 플러시 하지 않고 해당 구간의 해시 함수로 접근되지 않은 블록에 대해서만 비휘발성 메모리로 플러시 함으로써 비휘발성 메모리로의 쓰기 요청을 획기적으로

줄일 수 있다. 본 발명을 적용하는 경우, DRAM cache 내 전체 블록 중 5 내지 15% 정도만 플러시 되는 것으로 확인되었다.

[0113] 도 8은 본 발명의 일실시예에 따른 DRAM cache 내 개별 블록의 상태 천이 프로토콜 예시도로서, 도 7의 일부 블록만 플러시 하기 위한 프로토콜이다.

[0115] 본 발명의 일실시예에 따르면, 비트 카운터 콘트롤러(130)는 DRAM cache 내 개별 블록의 직전 해시 함수(Xp: Previous hash function)와 현재 해시 함수(Xc: Current hash function)를 기억하며, DRAM cache 내 일부 블록의 상태 천이 프로토콜은 다음과 같다.

[0117] i) dirty 상태

[0118] - DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, 해당 블록 내 데이터 Read 요청시, 해당 블록의 직전 해시 함수(Xp)나 현재 해시 함수(Xc)로 hit 하면(RdHit X{p//c}), 비트 카운터 콘트롤러(130)는 DRAM cache 내 해당 블록의 데이터 상태를 'dirty'로 유지한다(S810).

[0120] - DRAM cache 내 소정 블록의 데이터가 'dirty'이고, 해당 블록 내 데이터 Write 요청시, 해당 블록의 직전 해시 함수(Xp)나 현재 해시 함수(Xc)로 hit 하면(WrHit X{p//c}), 비트 카운터 콘트롤러(130)는 DRAM cache 내 해당 블록의 데이터 상태를 'dirty'로 유지한다(S820).

[0122] - DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, DRAM cache 내 블록의 데이터를 축출하고자 할 때(evict), 비트 카운터 콘트롤러(130)는 DRAM cache 내 해당 블록의 데이터를 플러싱 한다. 이때 비트 카운터 콘트롤러(130)는 DRAM cache 내 블록의 데이터 상태를 'dirty'에서 'invalid'로 천이시킨다(S830).

[0123] 즉, DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, DRAM cache 내 블록의 데이터를 축출하게 되면(evict), DRAM cache 내 해당 블록의 데이터를 비휘발성 메모리에 저장하는(flush) 것과 함께 DRAM cache 내 해당 블록의 데이터 상태를 초기화시킨다.

[0125] ii) clean 상태

[0126] - DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 해당 블록 내 데이터 Read 요청시, 해당 블록의 직전 해시 함수(Xp)나 현재 해시 함수(Xc)로 hit 하면(RdHit X{p//c}), 비트 카운터 콘트롤러(130)는 DRAM cache 내 해당 블록의 데이터 상태를 'clean'으로 유지한다(S840).

[0128] - DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 해당 블록 내 데이터 Write 요청시, 해당 블록의 직전 해시 함수(Xp)나 현재 해시 함수(Xc)로 hit 하면(WrHit X{p//c}), 비트 카운터 콘트롤러(130)는 DRAM cache 내 해당 블록의 데이터 상태를 'clean'에서 'dirty'로 천이시킨다(S850).

[0129] 즉, DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 해당 블록 내 데이터 Write 요청시, 직전 해시 함수(Xp)나 현재 해시 함수(Xc)로 hit 하면, DRAM cache 내 해당 블록의 데이터는 새로운 데이터가 기입되어 비휘발성 메모리의 동일 address 내 데이터와 불일치하게 되므로 비트 카운터 콘트롤러(130)는 DRAM cache 내 해당 블록의 데이터 상태를 'clean'에서 'dirty'로 천이시킨다.

[0131] - DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 해당 블록 내 데이터가 비휘발성 메모리로 축출되면(evict), 비트 카운터 콘트롤러(130)는 DRAM cache 내 해당 블록의 데이터 상태를 'clean'에서 'invalid'로 천이시킨다(S860).

[0132]

[0133] iii) invalid 상태

[0134] - DRAM cache 내 소정 블록의 데이터 상태가 'invalid'이고, 해당 블록 내 데이터 Write 요청시, 해당 블록의 직전 해시 함수(Xp)와 현재 해시 함수(Xc)가 모두 miss 이면(WrMiss X{p&&c}), 비트 카운터 콘트롤러(130)는 현재 해시 함수(Xc)를 이용하여 비휘발성 메모리의 해당 address의 데이터를 DRAM cache 내 동일 address로 읽어들이고(BusRd Xc), DRAM cache 내 해당 블록의 데이터 상태를 'invalid'에서 'dirty'로 천이시킨다(S870).

[0136] - DRAM cache 내 소정 블록의 데이터 상태가 'invalid'이고, 해당 블록 내 데이터 Read 요청시, 해당 블록의 직전 해시 함수(Xp)와 현재 해시 함수(Xc)가 모두 miss 이면(RdMiss X{p&&c}), 비트 카운터 콘트롤러(130)는 현재 해시 함수(Xc)를 이용하여 비휘발성 메모리의 해당 address의 데이터를 DRAM cache 내 동일 address로 읽어들이고(BusRd Xc), DRAM cache 내 해당 블록의 데이터 상태를 'invalid'에서 'clean'으로 천이시킨다(S880).

[0138] 도 9는 본 발명의 다른 실시예에 따른 DRAM cache 내 개별 블록의 상태 천이 프로토콜 예시도로서, 점진적으로 데이터를 축출하기 위한 프로토콜이다.

[0139] 본 발명의 다른 실시예에 따르면, 직전 해시 함수(Xp)를 이용하여 DRAM cache 내 개별 블록에 접근할 때 hit이 발생하면 다음과 같은 동작을 수행한다.

[0141] i) dirty 상태

[0142] - 해당 블록의 데이터 상태가 'dirty'이고, 해당 블록에 데이터를 쓰라는 Write 요청이면(WrHit Xp), 비트 카운터 콘트롤러(130)는 해당 블록의 데이터 상태를 'invalid'로 천이시키고, 해당 블록을 현재 해시 함수(Xc)로 접근하여 데이터 Write를 수행한다(Wr Xc)(S910).

[0144] - 해당 블록의 데이터 상태가 'dirty'이고, 해당 블록에 저장된 데이터를 읽으라는 Read 요청이면(RdHit Xp), 비트 카운터 콘트롤러(130)는 해당 블록의 데이터 상태를 'invalid'로 천이시키고, 비트 카운터 콘트롤러(130)는 해당 블록 대신 현재 해시 함수(Xc)로 접근하는 블록에 저장된 데이터의 Read를 수행한다(Move to Xc)(S915).

[0146] - DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, 해당 블록의 데이터를 축출하고자 할 때(evict), 비트 카운터 콘트롤러(130)는 DRAM cache 내 해당 블록의 데이터를 플러싱 한다. 이때 비트 카운터 콘트롤러(130)는 DRAM cache 내 블록의 데이터 상태를 'dirty'에서 'invalid'로 천이시킨다(S920).

[0147] 즉, DRAM cache 내 소정 블록의 데이터 상태가 'dirty'이고, DRAM cache 내 블록의 데이터를 축출하게 되면(evict), DRAM cache 내 해당 블록의 데이터를 비휘발성 메모리에 저장하는(flush) 것과 함께 DRAM cache 내 해당 블록의 데이터 상태를 초기화시킨다.

[0149] ii) clean 상태

[0150] - 해당 블록의 데이터 상태가 'clean'이고, 해당 블록에 데이터를 쓰라는 Write 요청이면(WrHit Xp), 비트 카운터 콘트롤러(130)는 해당 블록의 데이터 상태를 'invalid'로 천이시키고, 해당 블록을 현재 해시 함수(Xc)로 접근하여 데이터 Write를 수행한다(Wr Xc)(S930).

[0152] - 해당 블록의 데이터 상태가 'clean'이고, 해당 블록에 저장된 데이터를 읽으라는 Read 요청이면(RdHit Xp), 비트 카운터 콘트롤러(130)는 해당 블록의 데이터 상태를 'invalid'로 천이시키고, 비트 카운터 콘트롤러(130)는 해당 블록 대신 현재 해시 함수(Xc)로 접근하는 블록에 저장된 데이터의 Read를 수행한다(Move to

Xc)(S935).

[0154] - DRAM cache 내 소정 블록의 데이터 상태가 'clean'이고, 해당 블록 내 데이터가 비휘발성 메모리로 축출되면(evict), 비트 카운터 콘트롤러(130)는 DRAM cache 내 해당 블록의 데이터 상태를 'clean'에서 'invalid'로 천이시킨다(S940).

[0156] iii) invalid 상태

[0157] - DRAM cache 내 소정 블록이 invalid 상태에서, 해당 블록 내 데이터의 Write 요청시, 해당 블록에 직전 해시 함수(Xp)와 현재 해시 함수(Xc)로 모두 miss 이면(WrMiss X{p&&c}), 비트 카운터 콘트롤러(130)는 현재 해시 함수(Xc)를 이용하여 비휘발성 메모리의 해당 address의 데이터를 DRAM cache 내 동일 address로 읽어들이고(BusRd Xc), DRAM cache 내 해당 블록을 'invalid' 상태에서 'dirty' 상태로 천이시킨다(S950).

[0159] - DRAM cache 내 소정 블록이 invalid 상태에서, 해당 블록 내 데이터의 Read 요청시, 해당 블록에 직전 해시 함수(Xp)와 현재 해시 함수(Xc)로 모두 miss 이면(RdMiss X{p&&c}), 비트 카운터 콘트롤러(130)는 현재 해시 함수(Xc)를 이용하여 비휘발성 메모리의 해당 address의 데이터를 DRAM cache 내 동일 address로 읽어들이고(BusRd Xc), DRAM cache 내 해당 블록을 'invalid' 상태에서 'clean' 상태로 천이시킨다(S955).

[0161] 한편, DRAM cache 내 소정 블록의 데이터 상태가 'dirty' 에서 데이터의 Read 또는 Write 요청시 현재 해시 함수(Xc)로 hit 이면(RdHit Xc, WrHit Xc), 현 상태를 유지하고, 해당 요청을 수행한다(S960, S965).

[0162] 그리고, DRAM cache 내 소정 블록의 데이터 상태가 'clean' 에서 데이터의 Write 요청시 현재 해시 함수(Xc)로 hit 이면(WrHit Xc), 'clean' 상태를 'dirty' 상태로 천이시키고 해당 요청을 수행한다(S970).

[0163] 하지만, DRAM cache 내 소정 블록의 데이터 상태가 'clean' 에서 데이터의 Read 요청시 현재 해시 함수(Xc)로 hit 이면(RdHit Xc), 현 상태를 유지하고, 해당 요청을 수행한다(S975).

[0164]

[0165] 이상, 첨부된 도면을 참조하여 본 발명의 실시 예를 설명하였지만, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자는 본 발명이 그 기술적 사상이나 필수적인 특징을 변경하지 않고서 다른 구체적인 형태로 실시될 수 있다는 것을 이해할 수 있을 것이다. 그러므로 이상에서 기술한 실시 예에는 모든 면에서 예시적인 것이며 한정적이 아닌 것으로 이해해야만 한다.

부호의 설명

[0167] 110: 호스트 장치

120: 메인 콘트롤러

130: 비트 카운터 콘트롤러

131: 세트용 비트 카운터

133: 태그용 비트 카운터

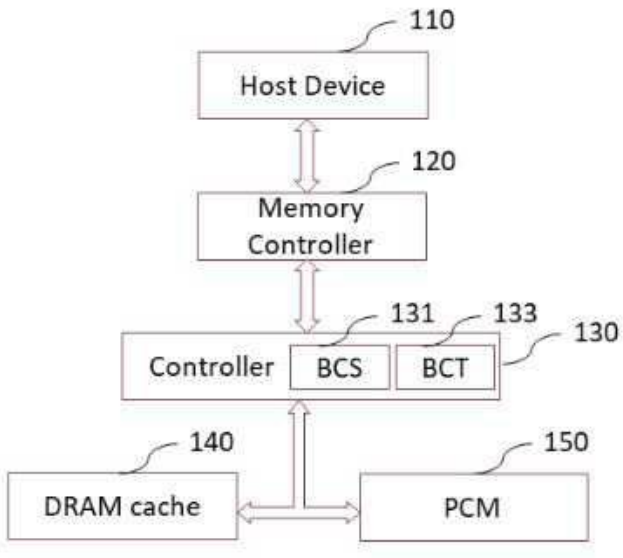
140: DRAM cache

150: 비휘발성 메모리

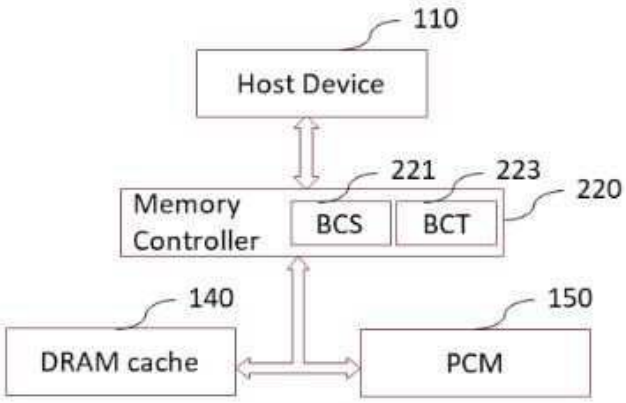
220: 메모리 콘트롤러

도면

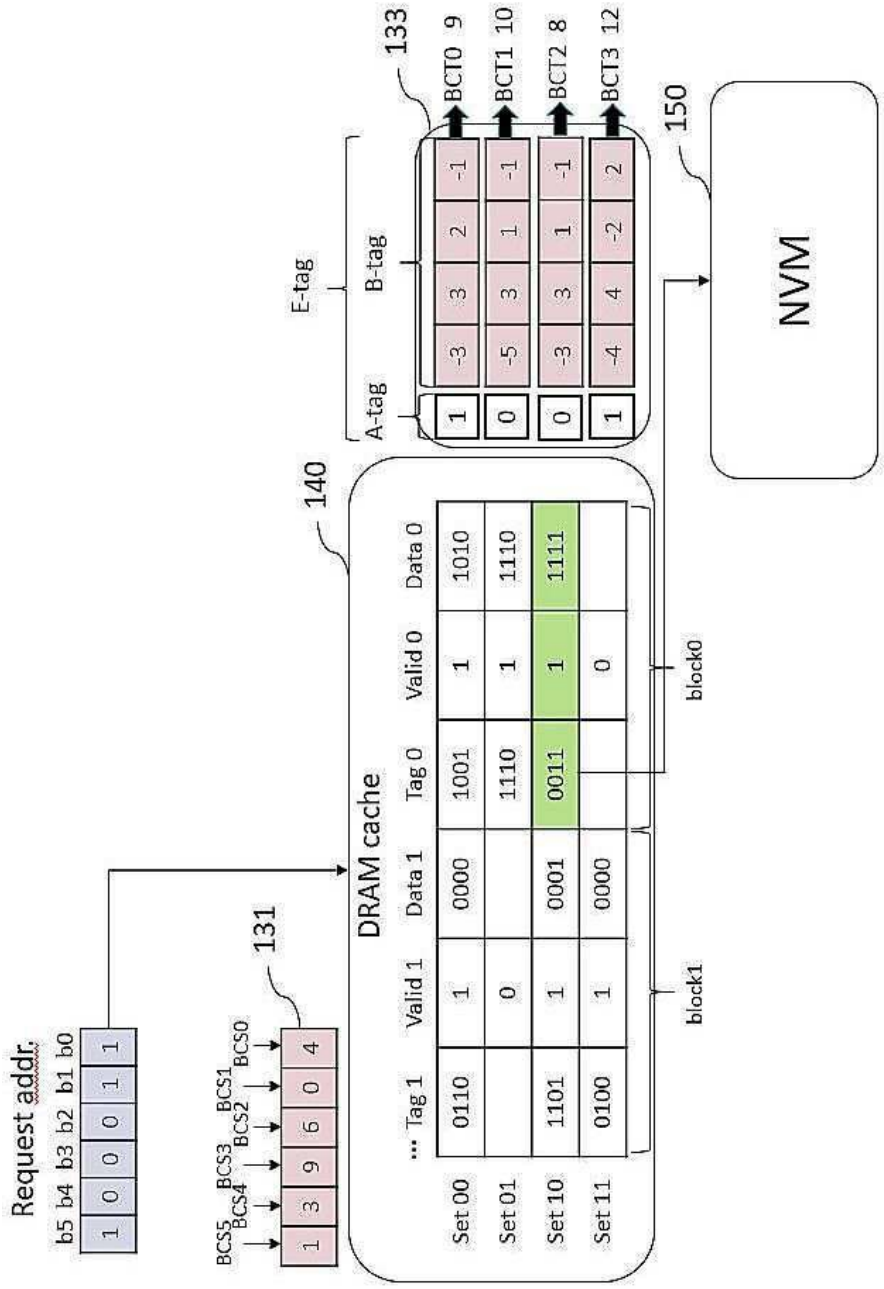
도면1


110
Host Device
120
Memory Controller
131
133
130
Controller
BCS
BCT
140
150
DRAM cache
PCM

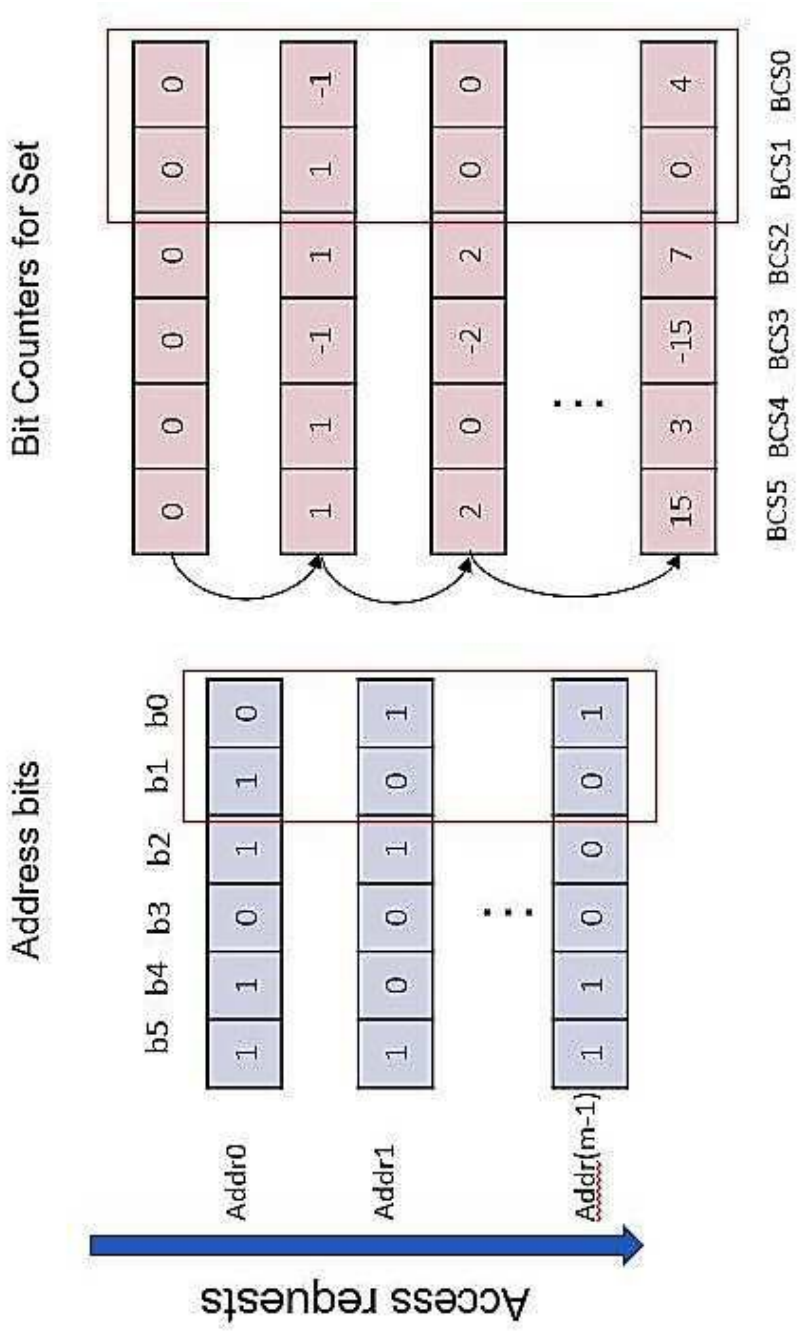
도면2


110
Host Device
221
223
220
Memory Controller
BCS
BCT
140
150
DRAM cache
PCM

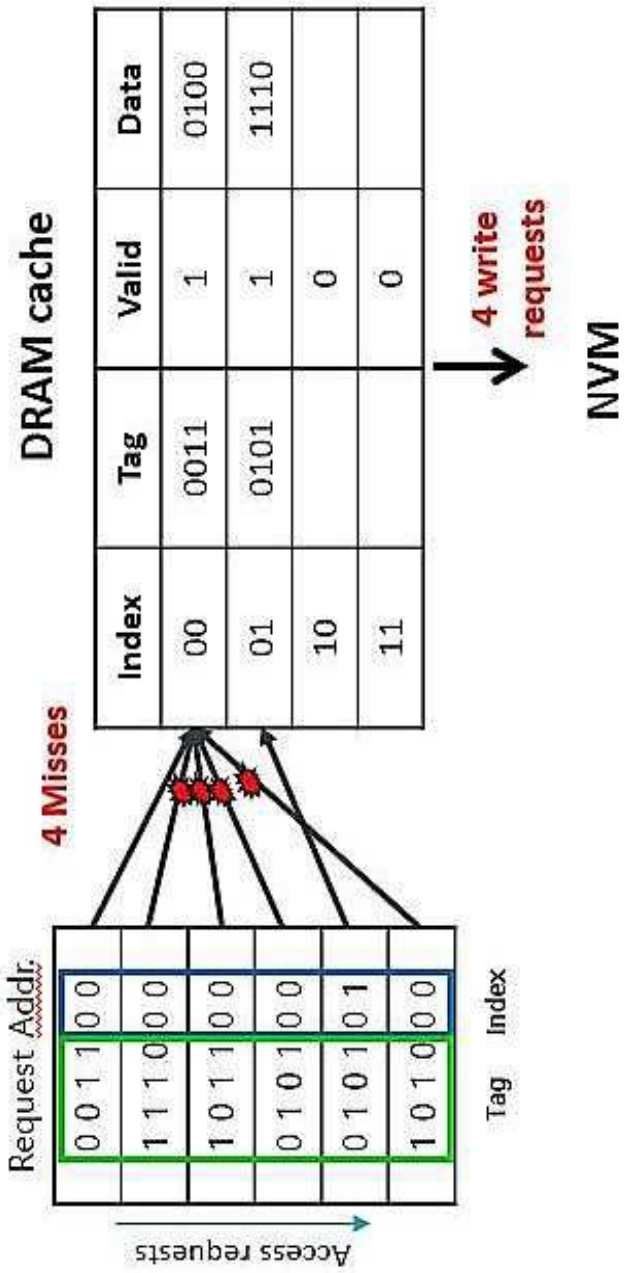
도면3


Request addr.
b5 b4 b3 b2 b1 b0
1 0 0 0 1 1
131
BCS5 BCS4 BCS3 BCS2 BCS1 BCS0
1 3 9 6 0 4
DRAM cache
140
... Tag 1 Valid 1 Data 1 Tag 0 Valid 0 Data 0
Set 00 0110 1 0000 1001 1 1010
Set 01 0 1110 1 1110
Set 10 1101 1 0001 0011 1 1111
Set 11 0100 1 0000 0 block0
block1
E-tag
A-tag
B-tag
133
BCT0 9
BCT1 10
BCT2 8
BCT3 12
1 -3 3 2 -1
0 -5 3 1 -1
0 -3 3 1 -1
1 -4 4 -2 2
150
NVM

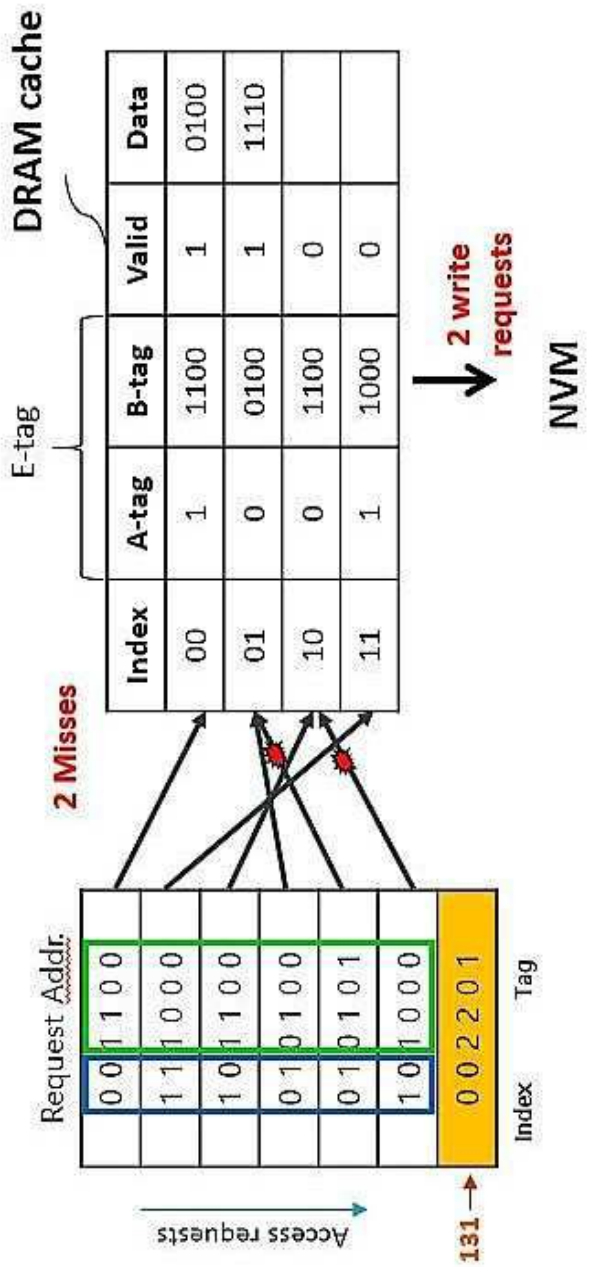
도면4


Bit Counters for Set
BCS0
BCS1
BCS2
BCS3
BCS4
BCS5
Address bits
b0
b1
b2
b3
b4
b5
Addr0
Addr1
Addr(m-1)
Access requests

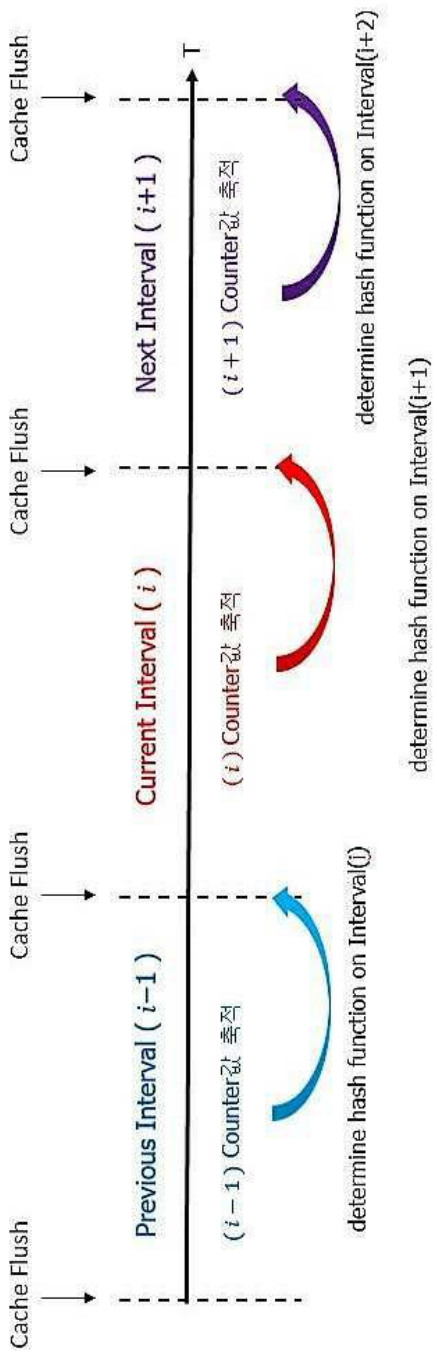
도면5a


DRAM cache
Index
Tag
Valid
Data
00
0011
1
0100
01
0101
1
1110
10
0
11
0
4 write requests
NVM
4 Misses
Request Addr.
0 0 1 1 0 0
1 1 1 0 0 0
1 0 1 1 0 0
0 1 0 1 0 0
0 1 0 1 0 1
1 0 1 0 0 0
Tag
Index
Access requests

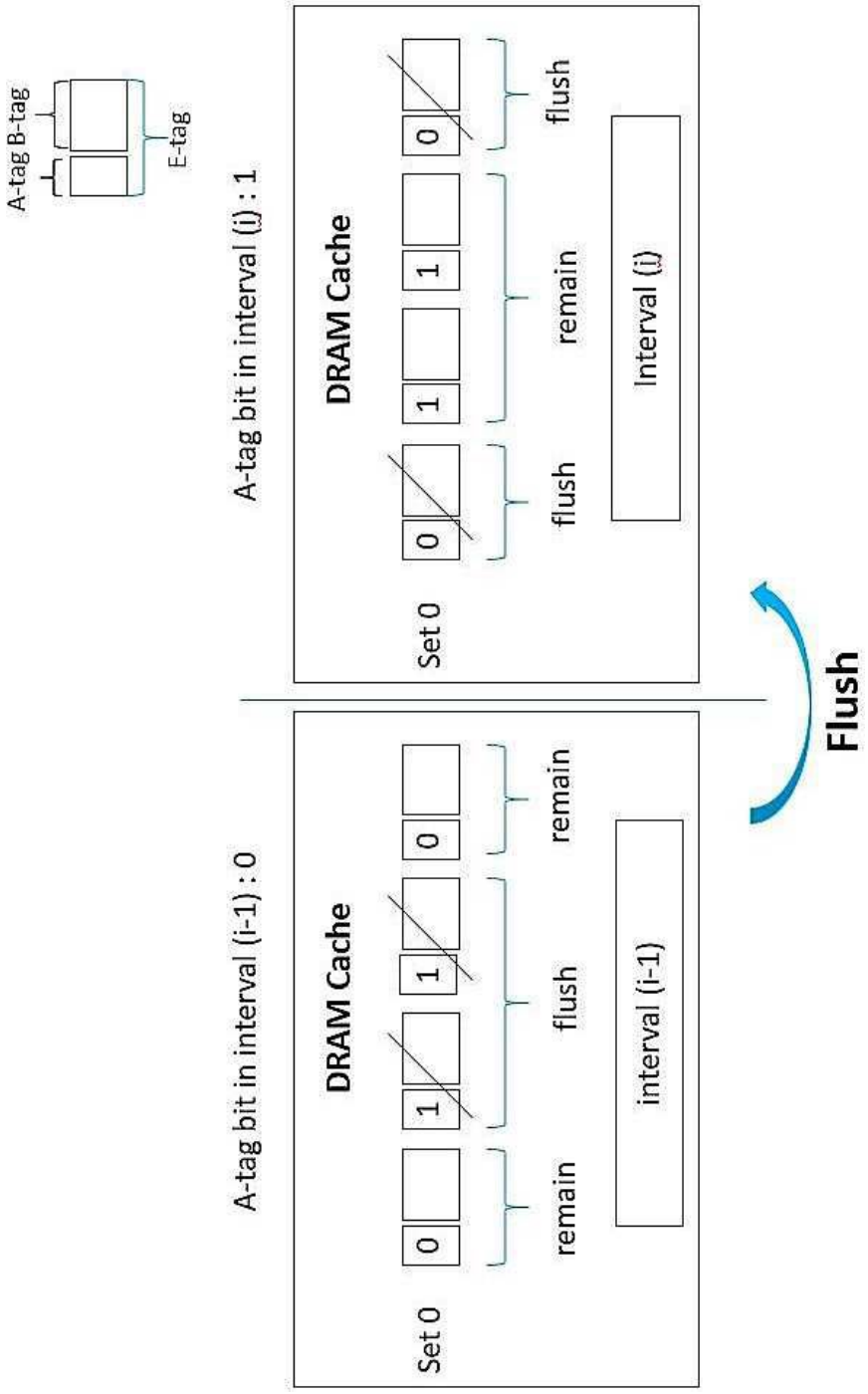
도면5b


DRAM cache
E-tag
Index
A-tag
B-tag
Valid
Data
00
1
1100
1
0100
01
0
0100
1
1110
10
0
1100
0
11
1
1000
0
2 write requests
NVM
2 Misses
Request Addr.
Tag
Index
131
Access requests

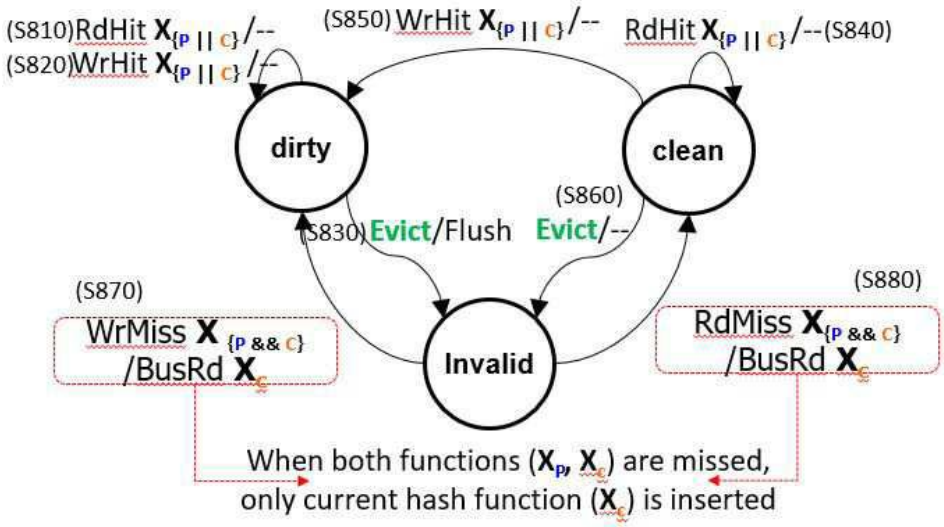
도면6


Cache Flush
Cache Flush
Cache Flush
Cache Flush
T
Previous Interval (i−1)
Current Interval (i)
Next Interval (i+1)
(i − 1) Counter값 측정
(i) Counter값 측정
(i + 1) Counter값 측정
determine hash function on Interval(i)
determine hash function on Interval(i+1)
determine hash function on Interval(i+2)

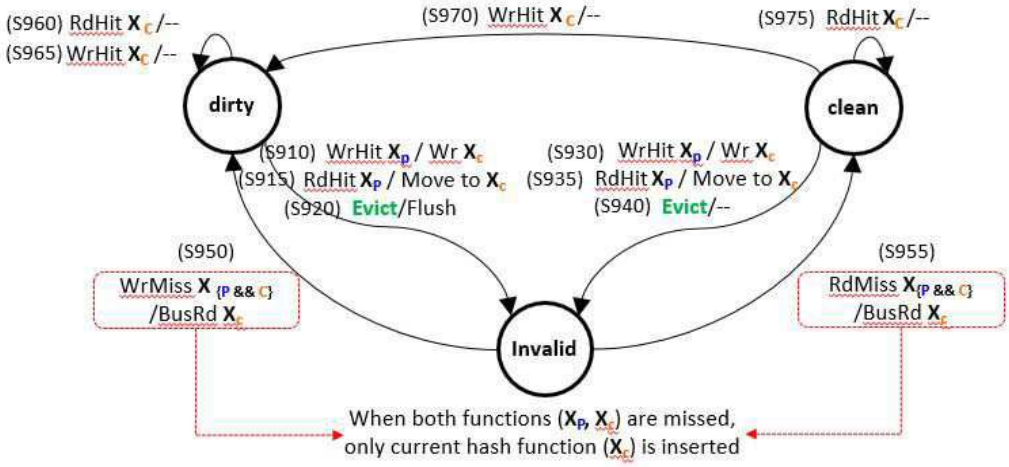
도면7


A-tag B-tag
E-tag
A-tag bit in interval (i) : 1
DRAM Cache
Set 0
0
1
1
0
flush
remain
flush
Interval (i)
Flush
A-tag bit in interval (i-1) : 0
DRAM Cache
Set 0
0
1
1
0
remain
flush
remain
interval (i-1)

도면8


(S810)RdHit X{P || C} /--
(S820)WrHit X{P || C} /--
(S850) WrHit X{P || C} /--
RdHit X{P || C} /--(S840)
dirty
clean
(S860)
(S830) Evict/Flush
Evict/--
(S870)
WrMiss X {P && C}
/BusRd XC
Invalid
(S880)
RdMiss X{P && C}
/BusRd XC
When both functions (XP, XC) are missed,
only current hash function (XC) is inserted

도면9


(S960) RdHit XC /--
(S965) WrHit XC /--
(S970) WrHit XC /--
(S975) RdHit XC /--
dirty
clean
(S910) WrHit XP / Wr XC
(S915) RdHit XP / Move to XC
(S920) Evict/Flush
(S930) WrHit XP / Wr XC
(S935) RdHit XP / Move to XC
(S940) Evict/--
(S950)
WrMiss X {P && C}
/BusRd XC
Invalid
(S955)
RdMiss X{P && C}
/BusRd XC
When both functions (XP, XC) are missed,
only current hash function (XC) is inserted