(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2009년06월08일
(11) 등록번호 10-0901551
(24) 등록일자 2009년06월01일

(51) Int. Cl.
G11C 16/02 (2006.01)
(21) 출원번호 10-2003-0075137
(22) 출원일자 2003년10월27일
심사청구일자 2008년10월22일
(65) 공개번호 10-2004-0038708
(43) 공개일자 2004년05월08일
(30) 우선권주장
60/421,910 2002년10월28일 미국(US)
(56) 선행기술조사문헌
US19935222109 A1
US19945297148 A1
US19955438573 A1
US19995860082 A1
전체 청구항 수 : 총 18 항

(73) 특허권자
샌디스크 코포레이션
미국 캘리포니아 95035 밀피타스 맥카시 블러바드 601
(72) 발명자
창,로버트씨.
미국94506캘리포니아단빌스탠톤코트10
콰와미,바만
미국95138캘리포니아샌어제이킬라니써클5899
(뒷면에 계속)
(74) 대리인
남상선

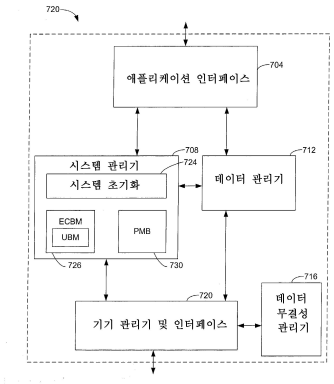
심사관 : 이경홍

(54) 비휘발성 메모리 시스템에서 블록 캐싱을 수행하는 방법 및 시스템

(57) 요 약

논리 블록과 관련된 업데이트 또는 새로운 데이터를 캐싱하는 방법 및 장치가 개기되어 있다. 본 발명의 일 실시예에 따라, 비휘발성 메모리의 블록에 관련된 콘텐츠를 처리하는 방법은 제1 논리 블록에 관련된 논리 페이지의 제1 그룹과 관련된 제1 세트의 콘텐츠를 얻는 단계, 상기 제1 세트의 콘텐츠를 메모리 영역에 기록하는 단계를 포함한다. 상기 제1 세트의 콘텐츠는 상기 메모리 영역으로부터 상기 제1 논리 블록에 맵되는 제2 물리 블록에 관련된 물리 페이지의 제1 그룹으로 기록된다. 일 실시예에서, 상기 메모리 영역은 RAM 캐시 및 물리 블록 캐시 중 하나이다.

대 표 도 - 도8



(72) 발명자

사벳-샤기,파시드

미국95138캘리포니아샌어제이스노우돈플레이스5634

리,핑

미국95051캘리포니아산타클라라록리지웨이2450

특허청구의 범위

청구항 1

비휘발성 메모리의 블록들과 연관되는 콘텐츠를 처리하기 위한 방법으로서, 상기 비휘발성 메모리는 메모리 시스템과 연관되고, 상기 방법은,

제 1 논리 블록을 제 1 물리 블록에 맵핑시키는 단계 - 상기 제 1 논리 블록은 다수의 논리 그룹들을 포함하고, 각각의 논리 그룹은 다수의 논리 페이지들을 포함하며, 상기 제 1 물리 그룹은 다수의 물리 그룹들을 포함하고, 각각의 물리 그룹은 다수의 물리 페이지들을 포함함 -;

상기 제 1 논리 블록의 제 1 논리 그룹과 연관되는 제 1 세트의 콘텐츠를 상기 제 1 물리 블록의 제 1 물리 그룹에 기록하는 단계;

상기 제 1 논리 그룹의 논리 페이지들과 연관되는 제 2 세트의 콘텐츠를 메모리 영역에 기록하는 단계;

상기 제 1 물리 블록의 상기 제 1 물리 그룹으로부터의 상기 제 1 세트의 콘텐츠와 상기 메모리 영역으로부터의 상기 제 2 세트의 콘텐츠를 제 2 물리 블록의 제 1 물리 그룹에 기록하는 단계 - 상기 제 2 물리 블록은 다수의 물리 그룹들을 포함하고, 각각의 물리 그룹은 다수의 물리 페이지들을 포함함 -; 및

상기 제 2 물리 블록을 상기 제 1 논리 블록에 맵핑시키는 단계

를 포함하는 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 2

제 1 항에 있어서,

상기 제 1 물리 블록을 삭제하는 단계; 및

상기 제 1 논리 블록으로부터 상기 제 1 물리 블록을 맵핑해제(unmapping)하는 단계를 더 포함하는, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 3

제 2 항에 있어서,

상기 메모리 영역으로부터 상기 제 2 세트의 콘텐츠를 삭제하는 단계를 더 포함하는, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 4

제 1 항에 있어서,

상기 제 1 세트의 콘텐츠를 상기 메모리 영역에 기록하는 단계는 상기 메모리 영역에 이전에 저장된 콘텐츠를 오버라이트하는 단계를 포함하는, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 5

제 4 항에 있어서,

상기 메모리 영역은 RAM 캐시인, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 6

제 1 항에 있어서,

상기 제 2 세트의 콘텐츠는 상기 제 1 논리 블록과 연관되는 업데이트를 포함하는, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 7

제 1 항에 있어서,

제 3 세트의 콘텐츠를 획득하는 단계

- 상기 제 3 세트의 콘텐츠는 상기 제 1 논리 블록의 제 2 논리 그룹과 연관되며, 상기 제 2 논리 그룹은 상기 제 1 논리 블록과 연관되는 다수의 제 2 논리 페이지들을 포함함 -;

상기 제 3 세트의 콘텐츠를 상기 메모리 영역에 기록하는 단계; 및

상기 제 3 세트의 콘텐츠를 상기 제 2 물리 블록의 제 2 물리 그룹에 기록하는 단계

를 더 포함하는, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 8

제 7 항에 있어서,

상기 제 1 세트의 콘텐츠를 기록하는 단계와 상기 제 3 세트의 콘텐츠를 상기 제 2 물리 블록의 상기 제 2 물리 그룹에 기록하는 단계는 상기 제 1 논리 블록에 대한 기록될 부가적인 콘텐츠가 현재 존재하지 않는다는 결정에 응답하여 수행되는, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 9

제 8 항에 있어서,

상기 제 1 논리 블록과 연관되는 기록될 콘텐츠가 현재 존재하지 않는다는 상기 결정은 제 2 논리 블록이 업데이트될 것이라고 결정하는 것을 포함하는, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 10

제 9 항에 있어서,

제 1 물리 블록을 삭제하는 단계; 및

상기 제 1 논리 블록으로부터 상기 제 1 물리 블록을 맵핑해제하는 단계를 더 포함하는, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 11

제 1 항에 있어서,

상기 메모리 영역은 제 3 물리 블록 또는 RAM 캐시 중 하나인, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 12

제 1 항에 있어서,

상기 비휘발성 메모리는 NAND 플래시 메모리인, 비휘발성 메모리의 블록들과 연관되는 콘텐츠 처리 방법.

청구항 13

비휘발성 메모리 시스템 내의 제 1 논리 블록과 연관되는 업데이트된 콘텐츠를 처리하기 위한 방법으로서, 상기 제 1 논리 블록은 제 1 물리 블록에 맵핑되고, 상기 제 1 논리 블록은 다수의 논리 그룹들을 포함하며, 각각의 논리 그룹은 다수의 논리 페이지들을 포함하고, 상기 제 1 물리 블록은 다수의 물리 그룹들을 포함하며, 각각의 물리 그룹은 다수의 물리 페이지들을 포함하고, 상기 방법은,

상기 제 1 논리 블록과 연관되는 제 1 업데이트를 수신하는 단계 - 상기 제 1 업데이트는 상기 제 1 논리 블록의 제 1 논리 그룹의 콘텐츠에 대한 업데이트이고, 상기 제 1 논리 그룹의 콘텐츠는 상기 제 1 물리 블록의 제 1 물리 그룹에 기록됨 -;

상기 제 1 업데이트를 캐시에 저장하는 단계;

상기 제 1 업데이트를 포함하는 상기 캐시의 콘텐츠를 제 2 물리 블록에 저장할지 여부를 결정하는 단계;

상기 캐시의 콘텐츠가 상기 제 2 물리 블록에 저장되어야 한다고 결정하는 것에 응답하여 상기 캐시의 콘텐츠를

상기 제 2 물리 블록에 저장하는 단계 - 상기 캐시의 콘텐츠를 저장하는 단계는 상기 제 2 물리 블록의 제 1 물리 그룹에 상기 제 1 업데이트를 저장하는 단계를 포함하고, 상기 제 2 물리 블록은 다수의 물리 그룹들을 포함하며, 각각의 물리 그룹은 상기 제 1 물리 그룹에 포함된 다수의 물리 페이지들을 포함함 -;

그 다음, 상기 캐시의 콘텐츠가 상기 제 2 물리 블록에 저장된 이후, 상기 제 2 물리 블록을 상기 제 1 논리 블록에 맵핑시키는 단계; 및

상기 제 1 논리 블록으로부터 상기 제 1 물리 블록을 맵핑해제하는 단계

를 포함하는, 비휘발성 메모리 시스템내의 제 1 논리 블록과 연관되는 업데이트된 콘텐츠 처리 방법.

청구항 14

제 13 항에 있어서,

상기 캐시의 콘텐츠를 상기 제 2 물리 블록에 저장할지 여부를 결정하는 단계는 제 2 논리 블록이 처리되어야 하는지를 결정하는 단계를 포함하는, 비휘발성 메모리 시스템내의 제 1 논리 블록과 연관되는 업데이트된 콘텐츠 처리 방법.

청구항 15

제 13 항에 있어서,

상기 제 1 업데이트를 저장하는 단계는 상기 제 1 물리 블록과 연관되는 이전에 저장된 콘텐츠의 적어도 일부분을 오버라이트하는, 비휘발성 메모리 시스템내의 제 1 논리 블록과 연관되는 업데이트된 콘텐츠 처리 방법.

청구항 16

제 13 항에 있어서,

상기 캐시는 RAM 캐시인, 비휘발성 메모리 시스템내의 제 1 논리 블록과 연관되는 업데이트된 콘텐츠 처리 방법.

청구항 17

제 13 항에 있어서,

상기 캐시의 콘텐츠의 적어도 일부가 상기 제 2 물리 블록에 저장된 이후 상기 제 1 물리 블록을 삭제하는 단계를 더 포함하는, 비휘발성 메모리 시스템내의 제 1 논리 블록과 연관되는 업데이트된 콘텐츠 처리 방법.

청구항 18

제 13 항에 있어서,

상기 캐시는 물리 블록 캐시인, 비휘발성 메모리 시스템내의 제 1 논리 블록과 연관되는 업데이트된 콘텐츠 처리 방법.

청구항 19

삭제

청구항 20

삭제

청구항 21

삭제

청구항 22

삭제

청구항 23

삭제

청구항 24

삭제

청구항 25

삭제

청구항 26

삭제

청구항 27

삭제

청구항 28

삭제

명 세 서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

<33> 본 발명은 대용량 디지털 데이터 저장 시스템에 관한 것이다. 보다 구체적으로, 본 발명은 논리 블록의 논리 페이지와 관련된 업데이트된 콘텐츠가 상기 논리 블록에 상응하는 물리 블록에 기록되기 전에 캐시되도록 하는 방법 및 장치에 관한 것이다.

<34> 플래시 메모리 저장 시스템과 같은 비휘발성 메모리 시스템의 사용은 상기와 같은 메모리 시스템의 콤팩트한 물리적 사이즈와 반복하여 재프로그램할 수 있는 능력 때문에 증가되고 있다. 플래시 메모리 저장 시스템의 상기 콤팩트한 물리적 사이즈는 점점 유행하는 기기에서 상기 저장 시스템의 사용을 용이하게 한다. 플래시 메모리 저장 시스템을 사용하는 기기는 디지털 카메라, 디지털 캠코더, 디지털 음악 플레이어, 휴대용 개인용 컴퓨터 및 위치 추적 장치 등을 포함한다. 플래시 메모리에 포함되어 있는 비휘발성 메모리를 반복하여 재프로그램할 수 있는 능력은 플래시 메모리 저장 시스템으로 하여금 사용되고 재 사용될 수 있도록 한다.

<35> 일반적으로, 플래시 메모리 저장 시스템은 플래시 메모리 카드와 플래시 메모리 팁 세트를 포함하고 있다. 플래시 메모리 칩 세트는 일반적으로 플래시 메모리 구성요소와 제어기 구성요소를 포함한다. 전형적으로, 플래시 메모리 칩 세트는 내장형 시스템으로 결합될 수 있도록 설치된다. 상기 결합 또는 호스트 시스템의 제작은 전형적으로 다른 구성요소들 뿐만 아니라 구성요소의 형태로 플래시 메모리를 가지며, 따라서 플래시 메모리와 다른 구성요소들을 호스트 시스템으로 결합한다.

<36> 플래시 메모리 시스템의 논리 블록은 전형적으로 물리 블록에 맵된다. 예를 들어, 32 논리 페이지를 포함하고 있는 논리 블록은 32 물리 페이지를 포함하고 있는 물리 블록에 맵될 수 있다. 상기 논리 블록과 관련된 새로운 또는 업데이트된 데이터가 상기 물리 블록에 기록될 때, 그리고 상기 데이터를 수용할 수 있는 물리 블록의 여유가 더 이상 없을 때, 추가의 물리 블록이 전형적으로 구해진다. 일단 상기 추가의 물리 블록이 구해지면, 상기 새로운 또는 업데이트된 데이터는 상기 최초 물리적 블록으로부터의 데이터와 함께 상기 추가의 물리 블록에 통합된다.

<37> 도1은 상기 업데이트된 콘텐츠를 수용하기 위해 구해진 물리 블록들과 업데이트된 콘텐츠를 가지고 있는 논리 블록을 표현하는 계통도이다. 시간(t1)에서, 논리 블록(202)은 새로운 데이터가 논리 블록(202)의 제3의 논리 페이지(204)에 포함되도록 업데이트된다. 제3의 논리 페이지(204)와 관련된 업데이트된 콘텐츠가 물리 블록

(212)에 기록될 수 없을 정도로 물리 블록(212)이 완전히 채워질 때, 새로운 물리 블록(222)이 구해진다. 제3 물리 페이지(214)의 콘텐츠를 제외하고, 물리 블록(212)의 모든 물리 페이지(214)의 콘텐츠는 본질적으로 제3 논리 페이지(204)와 관련된 상기 업데이트된 콘텐츠와 함께 물리 블록(222)에 기록될 수 있다. 결국, 물리 블록(222)은 시간(t1)에서의 논리 블록(202)과 관련된 가장 최근의 데이터를 포함한다. 일단 물리 블록(222)이 기록되면, 논리 블록(202)은 물리 블록(222)에 맵되며, 물리 블록(212)은 전형적으로 삭제된다. 업데이트된 콘텐츠를 물리 블록에 기록하는 종래의 프로세스는 도2를 참고하여 이하에서 설명된다.

<38> 시간(t2)에서, 물리 블록(222)으로 맵되는 논리 블록(202)은 제1 논리 페이지(204)와 관련된 콘텐츠가 새로운 또는 업데이트되도록 업데이트된다. 물리 블록(222)은 완전히 채워지거나 또는 제1 논리 페이지(204)와 관련된 콘텐츠를 포함하도록 업데이트될 수 없기 때문에, 새로운 물리 블록(232)이 얻어진다. 제1 물리 페이지(224)의 콘텐츠를 제외하고, 물리 블록(222)의 모든 물리 페이지(224)의 콘텐츠는 물리 블록(232)이 시간(t2)에서의 논리 블록(202)과 관련된 가장 최근의 데이터를 포함하도록 제1 논리 페이지(214)와 관련되어 있는 업데이트된 콘텐츠와 함께 물리 블록(232)으로 기록될 수 있다. 물리 블록(222)은 일반적으로 물리 블록(232)이 기록되고 논리 블록(202)에 맵된 후에 삭제된다.

<39> 다음으로 도2를 참고하여, 논리 블록과 관련된 페이지의 업데이트된 콘텐츠를 물리 블록에 기록하는 하나의 방법과 관련된 단계들이 설명될 것이다. 논리 블록과 관련된 업데이트된 페이지가 물리 블록에 기록되도록 하는 프로세스(250)는 단계(245)에서 시작되는데, 여기서 예를 들어 논리 블록'A"와 관련된 페이지 'X'가 업데이트된다. 논리 블록 A의 업데이트된 페이지 X는 전형적으로 페이지 X와 관련된 변화되는 콘텐츠를 포함하고 있다. 일단 논리 블록 A의 페이지 X가 업데이트되면, 논리 블록 A에 맵되는 또는 이에 관련되어 잇는 새로운 물리 블록 'B'가 단계(258)에서 얻어진다. 새로운 물리 블록 B는 논리 블록 A와 현재 관련되어 있는 물리 블록이 상기 업데이트된 페이지 X의 콘텐츠가 기록될 수 있는 가용 여유를 가지고 있지 않을 때 얻어질 수 있다. 당업자가 이해할 수 있는 것과 같이, 새로운 물리 블록 B은 여유 블록 풀로부터 얻어질 수 있다.

<40> 단계(262)에서, 새로운 물리 블록 B가 얻어진 후에, 논리 블록 A와 현재 관련되어 있는 물리 블록 A의 이전 콘텐츠는 논리 블록 A의 페이지 X의 업데이트된 콘텐츠와 함께 물리 블록 B에 통합된다. 달리 말하면, 논리 블록 A의 페이지 X와 관련되어 있는 콘텐츠를 제외하고, 물리 블록 A의 모든 콘텐츠는 논리 블록 A의 페이지 X의 업데이트된 콘텐츠와 함께 논리 블록 B에 기록된다. 이와 같이, 물리 블록 B는 논리 블록 A와 관련된 가장 최근의 콘텐츠를 포함한다.

<41> 콘텐츠들이 물리 블록 B에 기록된 후에, 물리 블록 A는 단계(266)에서 삭제되며, 단계(270)에서 논리 블록 A로부터 분리된다. 일단 물리 블록 A가 삭제되고 논리 블록 A로부터 분리되면, 물리 블록 A는 전형적으로 여유 블록으로 사용될 수 있으며, 논리 블록과 관련된 업데이트된 페이지가 물리 블록에 기록되도록 하는 프로세스가 완료된다.

<42> 전형적으로, 콘텐츠를 물리 블록에 기록하는 프로세스와 물리 블록의 콘텐츠를 삭제하는 프로세스는 시간이 소요되며, 상당한 계산 자원을 소모한다. 종종, 논리 블록의 논리 페이지가 업데이트되는 시간에, 하나의 물리 블록은 또 다른 물리 블록이 삭제되는 동안 기록된다. 이와 같이, 업데이트되는 많은 페이지가 있을 때, 블록의 일정한 삭제, 새로운 블록을 얻는 것 및 데이터를 새로운 블록에 기록하는 것을 가지고 있다. 일정하게 블록에 기록하는 것과 블록을 삭제하는 것은 메모리 시스템의 성능을 상대적으로 상당하게 감소시킨다.

<43> 따라서, 필요한 것은 논리 페이지 업데이트에 응답하여 발생하는 기록 및 삭제 프로세스의 회수를 줄이는 방법 및 장치에 관한 것이다. 특히, 바라는 것은 논리 블록 도메인의 업데이트에 상응하는 물리 블록 도메인의 업데이트를 수행하는 것과 관련된 기록 및 삭제 작동 용량을 물리 블록 도메인의 업데이트를 수행하는 능력 감소 없이 줄이는 시스템이다.

발명이 이루고자 하는 기술적 과제

<44> 본 발명은 논리 블록과 관련되어 있는 새로운 데이터 또는 업데이트를 캐싱하는 방법 및 시스템에 관한 것이다. 본 발명의 일 측면에 상응하여, 비휘발성 메모리의 블록에 관련된 콘텐츠를 처리하는 방법은 제1 논리 블록과 관련되어 있는 논리 페이지의 제1 그룹과 관련된 제1 세트의 콘텐츠를 얻는 단계 및 상기 제1 세트의 콘텐츠를 메모리 영역에 기록하는 단계를 포함한다. 상기 제1 세트의 콘텐츠는 상기 메모리 영역으로부터 상기 제1 논리 블록으로 맵되는 제2 물리 블록과 관련된 제1 논리 페이지 그룹으로 기록된다. 일 실시예에서, 상기 메모리 영역은 RAM 캐시와 물리 블록 캐시 중 하나이다.

<45> 또 다른 실시예에서, 상기 제1 물리 블록은 제2 세트의 콘텐츠를 포함하며, 상기 방법은 상기 제2 세트의 콘텐

츠의 적어도 일부를 상기 제1 세트의 콘텐츠와 상기 제2 물리 블록에 기록하는 단계를 포함한다. 상기 실시예에서, 상기 제2 세트의 콘텐츠는 상기 제1 물리 블록으로부터 삭제될 수 있으며, 상기 제1 물리 블록은 맵되지 않거나 또는 본질적으로 상기 제1 논리 블록으로부터 분리될 수 있다.

<46> 논리 블록과 관련된 업데이트된 콘텐츠일 수 있는 새로운 콘텐츠를 캐싱하는 것은 논리 블록과 관련되어 있는 모든 새로운 콘텐츠가 프로세스되기까지 임시적으로 저장될 수 있도록 한다. 프로세스되어야 하는 모든 새로운 콘텐츠가 프로세스될 때, 상기 새로운 콘텐츠 중 적어도 일부는 상기 논리 블록과 관련된 가장 최근의 콘텐츠가 새로운 물리 블록으로 기록되도록 상기 논리 블록에 맵되는 최초 물리 블록의 적절한 콘텐츠와 함께 복사될 수 있다. 캐싱 프로세스를 사용함으로써, 업데이트 또는 논리 블록과 관련되어 있는 새로운 추가를 처리하는 것과 관련된 기록 및 판독 작동 회수는 일반적으로 줄어들 수 있다. 따라서, 캐싱 프로세스를 사용하는 전체 메모리 시스템의 성능은 향상될 수 있다.

<47> 본 발명의 또 다른 측면에 상응하여, 비휘발성 메모리 시스템에서 제1 논리 블록과 관련된 업데이트된 콘텐츠를 처리하는 방법은 제1 논리 블록과 관련된 제1 업데이트를 수신하는 단계, 상기 제1 업데이트를 캐시에 저장하는 단계 및 언제 캐시의 콘텐츠를 제2 물리 블록에 정장하여야 하는지를 결정하는 단계를 포함한다. 상기 방법은 또한 상기 캐시의 콘텐츠가 상기 제2 물리 블록에 저장되어야 한다고 결정될 때 제1 업데이트를 포함하는 캐시 콘텐츠를 상기 제2 물리 블록에 저장하는 단계를 포함한다. 상기 제2 물리 블록은 상기 제1 논리 블록에 맵될 수 있으며, 상기 제1 물리 블록은 상기 캐시의 콘텐츠가 상기 제2 물리 블록에 저장된 후에 상기 제1 논리 블록으로부터 맵되지 않을 수 있다.

<48> 일 실시예에서, 상기 캐시의 콘텐츠를 상기 제2 물리 블록에 저장할 것인지를 결정하는 단계는 언제 제2 논리 블록이 프로세스되어야 하는지를 결정하는 단계를 포함한다. 언제 상기 제2 논리 블록이 프로세스되어야 한다는 것이 결정될 때, 상기 캐시의 콘텐츠는 상기 제2 물리 블록으로 저장된다. 또 다른 실시예에서, 상기 캐시의 콘텐츠는 상기 제1 물리 블록과 관련된 적어도 일부의 콘텐츠를 포함한다.

<49> 본 발명의 다른 특징은 이하에서의 여러 도면들과 자세한 설명을 읽은 후에 보다 명확해질 것이다.

발명의 구성 및 작용

<50> 물리 블록에 기록하는 프로세스와 물리 블록을 삭제하는 프로세스는 상대적으로 시간을 소모하며 전체적으로 상당한 계산을 사용한다. 따라서, 기록과 삭제가 자주 발생할 때, 물리 블록을 포함하고 있는 시스템의 전체 성능은 역효과를 가질 수 있다. 논리 블록의 논리 페이지가 업데이트되는 각각의 시간에, 논리 블록에 맵되는 제1 물리 블록이 완전히 채워지면, 상기 제1 물리 블록의 업데이트된 그리고 일정한 콘텐츠는 상기 제1 물리 블록이 삭제되면서 새로운 물리 블록으로 기록될 수 있다. 논리 블록에 대해 업데이트되는 많은 페이지가 있다면, 물리 블록을 일정하게 삭제하는 단계, 새로운 물리 블록을 얻는 단계 및 데이터를 새로운 물리 블록으로 기록하는 단계를 가지고 있다. 물리 블록들이 효율적으로 일정하게 기록되거나 또는 삭제될 때, 블록들을 포함하고 있는 메모리 시스템의 성능은 저하될 수 있다.

단일 논리 블록의 논리 페이지들이 일정하게 업데이트되는 경우에, 논리 블록에 상응하는 물리 블록을 삭제하고 이에 기록하는데 소요되는 전체 시간의 양은 상당하다. 특히, 논리 블록에 있는 단일 논리 페이지가 반복하여 업데이트될 때, 논리 페이지에 대한 각각의 업데이트의 "물리 블록 도메인"에서 업데이트를 수행하는 것은 상기 논리 블록 도메인이 일정하게 변화할 때는 특히 쓸모 없는 일인데, 이는 물리 블록 도메인의 업데이트는 반복하여 재 기록되기 때문이다. 논리 블록에 대한 모든 업데이트가 완료되는 시간까지 물리 블록 업데이트를 기다리는 것은 기록 및 삭제 동작의 수를 줄일 수 있도록 한다. 따라서, 메모리 시스템의 성능은 개선될 수 있다.

<51> 논리 블록으로 업데이트를 캐시하거나 또는 상기 논리 블록과 관련된 새로운 데이터를 캐시하는 캐싱 프로세스를 사용하는 것은 논리 블록으로의 모든 업데이트 또는 논리 블록으로 새로운 데이터의 추가가 일정 시간동안에 완료될 때까지 일시적으로 업데이트가 저장될 수 있도록 한다. 일단 다른 논리 블록이 프로세스된 후에 또는 상기 캐시가 완전히 채워질 때까지 상기 논리 블록으로 추가되는 새로운 데이터 또는 추가의 업데이트가 더 이상 존재하지 않는다는 것을 효율적으로 알 수 있으면, 적어도 일부의 캐시된 업데이트 및 새로운 데이터는 논리 블록에 맵되는 최초 물리 블록의 적절한 콘텐츠와 함께 복사될 수 있으며, 따라서 상기 논리 블록과 관련된 가장 최근의 콘텐츠가 새로운 물리 블록에 기록된다. 상기 새로운 물리 블록이 기록될 때 상기 최초 물리 블록이 삭제된다. 결국, 다수의 업데이트 또는 새로운 추가의 처리와 관련된 기록 작동과 판독 작동의 수는 줄어들 수 있다. 이하에서, 보다 명확하게 설명하기 위해, 업데이트를 일반적으로 새로운 콘텐츠를 포함한다. 유사하게, 블록에서 콘텐츠를 업데이트하는 것은 일반적으로 새로운 콘텐츠를 상기 블록에 추가하는 단계를 포함한다.

<52> 시스템의 소프트웨어 요구가 RAM이 사용되는 메모리일 때, 캐시는 RAM 캐시일 수 있다. RAM 캐시의 사용은 전형적으로 기록 프로세스가 효율적으로 발생할 수 있도록 한다. 그러나, 일 실시예에서, 만약 충분한 RAM 가용양이 존재하지 않으면, 캐시는 블록 캐시일 수 있다. 즉, 캐시는 물리 블록이다.

<53> 플래시 메모리 시스템 또는 일반적으로 비휘발성 메모리 소자는 업데이트된 데이터를 물리 블록에 기록하기 전에 업데이트된 데이터를 일시적으로 캐시하는 블록 캐시 또는 RAM 캐시를 사용하는 이로움이 있다. 전형적으로, 플래시 메모리 시스템은 호스트 시스템이 데이터를 상기 플래시 메모리에 기록하거나 또는 상기 플래시 메모리부터 판독하도록 호스트 시스템과 결합하여 사용된다. 그러나, 일정한 플래시 메모리 시스템은 내장형 플래시 메모리와 도3c에서 설명되는 것과 같이 내장된 상기 플래시 메모리를 위한 제어기로 작동하는 호스트를 실행하는 소프트웨어를 포함하고 있다. 도3a를 참고하여, 예를 들어 콤팩트 플래시 메모리 카드와 같은 비휘발성 메모리를 포함하고 있는 일반 호스트 시스템이 설명된다. 호스트 또는 컴퓨터 시스템(100)은 일반적으로 마이크로프로세서(108), RAM(112) 및 입력/출력 회로(116)기 통신하도록 하는 시스템 버스(104)를 포함하고 있다. 호스트 시스템(100)은 예를 들어 디스플레이 기기 및 네트워킹 기기와 같은 다른 구성요소를 일반적으로 포함하는데, 도시되어 있지는 않다.

<54> 일반적으로, 호스트 시스템(100)은 정지 이미지 정보, 오디오 정보 및 비디오 이미지 정보를 포함하는 그러나 이에 한정되지 않는 정보를 캡처링할 수 있다. 상기 정보는 실시간으로 캡처될 수 있으며, 무선 방식으로 호스트 시스템(100)으로 송신될 수 있다. 호스트 시스템(100)은 어느 일정한 시스템일 수 있지만, 호스트 시스템(100)은 디지털 카메라, 비디오 카메라, 셀룰러 통신 기기, 오디오 플레이어 또는 비디오 플레이어와 같은 시스템이다. 그러나 상기 호스트 시스템(100)은 데이터 또는 정보를 저장하고 데이터 또는 정보를 검색하는 어느 일정한 시스템일 수 있다.

<55> 호스트 시스템(100)은 또한 데이터를 캡처하거나 또는 데이터를 검색하기만 하는 시스템일 수 있다. 즉, 호스트 시스템(100)은 일 실시예에서, 데이터를 저장하는 전용 시스템일 수 있으며, 데이터를 판독하는 전용 시스템일 수 있다. 예로서, 호스트 시스템(100)은 데이터 기록 또는 판독만을 위해 구성되는 메모리 기록기일 수 있다. 대안적으로, 호스트 시스템(100)은 데이터를 판독하고 검색하며 캡처하지 않도록 구성된 MP3 플레이어와 같은 기기일 수 있다.

<56> 일 실시예에서, 제거할 수 있는 비휘발성 메모리 소자와 같은 비휘발성 메모리 소자(120)는 정보를 저장하기 위해 버스(104)와 인터페이스하도록 구성된다. 선택적인 인터페이스 블록(130)은 비휘발성 메모리 소자(102)로 하여금 버스(104)와 간접적으로 인터페이스하도록 한다. 존재할 때, 입력/출력 회로 블록(116)은 당업자가 이해할 수 있는 것과 같이, 버스(104)의 부하를 줄이는데 사용된다. 비휘발성 메모리 소자(120)는 비휘발성 메모리(124)와 선택적인 메모리 제어 시스템을 포함한다. 일 실시예에서, 비휘발성 메모리 소자(120)는 단일 칩 또는 다이 상에서 구현될 수 있다. 대안적으로, 비휘발성 메모리 소자(120)는 칩 세트를 형성하여 함께 비휘발성 메모리 소자(120)로서 사용될 수 있는 다수의 개별 컴포넌트들, 또는 다중-칩 모듈 상에서 구현된다. 비휘발성 메모리 소자(120)의 일 실시예는 이하에서 도3b를 참고로 보다 자세히 설명될 것이다.

<57> NAND 플래시 메모리와 같은 예를 들어 플래시 메모리와 같은 비휘발성 메모리(124)는 요구되는 것과 같이 액세스되어 판독될 수 있도록 데이터를 저장하도록 구성된다. 비휘발성 메모리(124)에 저장되어 있는 데이터는 또한 적절히 삭제될 수 있으며, 비휘발성 메모리(124)의 일정한 데이터는 삭제되지 않을 수 있다는 것을 이해하여야 한다. 데이터를 저장하고 판독하고 그리고 삭제하는 프로세스는 일반적으로 메모리 제어 시스템(128)에 의해 제어되며, 메모리 제어 시스템(128)이 존재하지 않을 때는 마이크로프로세서(108)에 의해 수행되는 소프트웨어로 제어된다. 비휘발성 메모리(124)의 작동은 비휘발성 메모리(124)의 섹션이 동일하게 닳도록 하여 상기 비휘발성 메모리(124)의 수명이 최대가 되도록 관리될 수 있다.

<58> 비휘발성 메모리 소자(120)는 일반적으로 추가적인 메모리 제어 시스템(128), 즉 제어기를 포함하도록 설명되어 있다. 종종, 비휘발성 메모리 소자(120)는 비휘발성 메모리(124)를 위한 분리된 칩과 메모리 제어 시스템(128), 즉 제어기 기능을 포함할 수 있다. 예로써, PC 카드, 콤팩트 플래시 카드, 멀티미디어 카드 및 안전 디지털 카드를 포함하는, 그러나 이에 한정되지 않는 비휘발성 메모리 소자는 분리된 칩에서 구현될 수 있는 제어기를 포함하는 반면, 다른 비휘발성 메모리 소자는 분리된 칩에서 구현되는 제어기를 포함하지 않을 수 있다. 비휘발성 메모리 소자(120)가 분리된 메모리와 제어기 칩을 포함하지 않은 일 실시예에서, 상기 메모리와 제어기 기능은 당업자가 이해할 수 있는 것과 같이, 단일의 칩에 통합될 수 있다. 대안적으로, 메모리 제어 시스템(128)의 기능은 비휘발성 메모리 소자(120)가 메모리 제어기(128)를 포함하지 않는 일 실시예에서 마이크로프로세서(108)에 의해 제공될 수 있다.

<59> 도3b를 참고로, 비휘발성 메모리 소자(120)는 본 발명의 일 실시예에 상응하여 이하에서 보다 자세히 설명된다. 상기 설명한 것과 같이, 비휘발성 메모리 소자(120)는 비휘발성 메모리(124)를 포함하며, 메모리 제어 시스템(128)을 포함할 수 있다. 일 실시예에서, 비록 메모리(124)가 내장형 NAND 기기일 때, 비휘발성 메모리 소자(120)는 제어 시스템(128)을 포함하지 않을 수 있지만, 메모리(124)와 제어 시스템(128) 또는 제어기는 비휘발성 메모리 소자의 주요 구성요소이다. 메모리(124)는 반도체 기판 위에 형성된 메모리 셀들의 어레이일 수 있는데, 여기서 데이터의 하나 이상의 비트는 메모리 셀들의 개별적인 저장 구성요소 상의 2개 또는 그 이상의 레벨 중 하나를 저장함으로써 개별적인 메모리 셀에 저장된다. 비휘발성 EEPROM은 상기 시스템을 위한 일반적인 타입의 메모리이다.

<60> 비휘발성 메모리 소자(120)에 존재할 때, 제어 시스템(128)은 버스(15) 상에서 데이터를 저장하는 메모리 시스템을 사용하고 있는 호스트 컴퓨터 또는 다른 시스템과 통신한다. 버스(15)는 일반적으로 도 3a의 버스(104)의 일부이다. 제어 시스템(128)은 또한 메모리124)의 작동을 제어하는데, 메모리 셀 어레이(11)를 포함하며, 호스트에 의해 제공된 데이터를 기록하고 호스트에 의해 요구되는 데이터를 판독하고 작동하는 메모리(124)에서 여러 하우스 키핑(house keeping) 기능을 수행한다. 제어 시스템(128)은 일반적으로 관련된 비휘발성 소프트웨어 메모리, 여러 논리 회로 등을 가지고 있는 범용 마이크로프로세서를 포함하고 있다. 하나 이상의 상태 기계가 특정 루틴의 성능을 제어하기 위해 포함될 수 있다.

<61> 일 실시예에서, 메모리 셀 어레이(11)는 어드레스 디코더(17)를 통해 제어 시스템(128) 또는 마이크로프로세서(108)에 의해 어드레스될 수 있다. 상기 실시예에서, 디코더(17)는 데이터를 상기 제어 시스템(128)에 의해 어드레스되는 메모리 셀의 그룹에 프로그램하거나 상기 셀의 그룹으로부터 판독하거나 삭제하기 위해 어레이(11)의 비트 라인과 게이트에 정확한 전압을 가한다. 추가적인 회로(19)가 어드레스된 셀의 그룹으로 프로그램되는 데이터에 의존하여 어레이 엘리먼트에 적용되는 전압을 제어하는 프로그래밍 드라이버를 포함할 수 있다. 회로(19)는 또한 감지 증폭기 및 어드레스된 메모리 셀의 그룹으로부터 데이터를 판독하는데 필요한 다른 회로를 포함할 수 있다. 어레이(11)로 프로그램되는 데이터 또는 어레이(11)로부터 최근에 판독된 데이터는 비록 데이터가 다른 버퍼 메모리(미도시)에 저장될 수 있지만, 데어 시스템(128)과 관련되어 있는 버터 메모리(21)에 저장될 수 있다. 제어 시스템(128)은 또한 명령 및 상태 데이터를 일시적으로 저장하는 여러 레지스터를 포함할 수 있다.

<62> 어레이(11)는 종종 블록 0-N 개의 메모리 셀들로 분리된다. 플래시 EEPROM 시스템에 공통되는 것과 같이, 상기 블록은 가장 작은 삭제 유닛일 수 있다. 즉, 각 블록은 함께 삭제되는 최소한의 메모리 셀의 수를 포함하도록 구성될 수 있다. 각각의 블록은 전형적으로 다수의 페이지로 분리된다. 당업자가 이해할 수 있는 것과 같이, 페이지는 프로그램되는 가장 작은 유닛으로 생각될 수 있다. 즉, 기본적인 프로그램밍 작동은 데이터를 메모리 셀의 최소한 하나의 페이지로 기록하거나 또는 판독하는 것이다. 데이터의 하나 이상의 섹터들은 각 페이지에 저장될 수 있다. 도3b에 도시되어 있는 것과 같이, 하나의 섹터는 사용자 데이터와 오버헤드 데이터를 포함하고 있다. 오버헤드 데이터는 전형적으로 상기 섹터의 사용자 데이터로부터 계산되는 에러 수정 코드(ECC)를 포함한다. 상기제어 시스템(15)의 부분(23)은 데이터가 어레이(11)로 프로그램될 때 ECC를 계산하며, 데이터가 어레이(11)로부터 판독될 때 상기 ECC를 체크한다. 대안적으로, 상기 ECC는 그들이 포함되어 있는 사용자 데이터보다는 서로 다른 페이지 또는 서로 다른 블록에 저장된다.

<63> 사용자 데이터 섹터는 전형적으로 자기 디스크 드라이브의 섹터 사이즈에 상응하는 512 바이트이다. 오버헤드 데이터는 비록 다른 수의 바이트를 가질 수 있지만 전형적으로 추가적인 16바이트이다. 데이터의 하나의 섹터는 각각의 페이지에 있지만, 2개 이상의 섹터들이 페이지를 형성할 수 있다. 다른 수의 페이지들이 블록을 형성한다. 예로써, 블록은 8페이지에서 512, 1024 또는 그 이상의 페이지로부터 형성될 수 있다. 상기 블록의 수는 종종 메모리 시스템에 대한 바람직한 데이터 저장 용량을 제공하도록 선택된다. 어레이(11)는 전형적으로 수 개의 서브 어레이(미도시)로 분리되는데, 각각은 블록들의 일정 부분을 포함하며, 여러 메모리 작동을 실행하는 병렬처리 정도를 강화하기 위해 서로에게 어느 정도 독립적으로 작동된다. 다수의 서브 어레이를 사용하는 예는 미국 특허 제 5,890,192에 설명되어 있는데, 이하에서 참고로 통합되어 있다.

<64> 일 실시예에서, 비휘발성 메모리는 시스템, 예를 들어 호스트 시스템에 내장되어 있다. 도3c는 내장형 비휘발성 메모리를 포함하고 있는 호스트 시스템의 계통도이다. 호스트 또는 컴퓨터 시스템(150)은 일반적으로 호스트 시스템(150)의 다른 구성요소들 중에서 마이크로프로세서(158), RAM(162) 및 입력/출력 회로(166)가 통신하도록 하는 시스템 버스(154)를 포함한다. 비휘발성 메모리(174), 예를 들어 플래시 메모리는 정보가 호스트 시스템(150)안에 저장될 수 있도록 한다. 인터페이스(180)는 비휘발성 메모리(174)와 버스(154)에 제공되어 정보

가 상기 비휘발성 메모리(174)로 기록되거나 이로부터 판독되도록 한다.

<65> 비휘발성 매모리(174)는 비휘발성 메모리(174)를 제어하도록 구성된 소프트웨어 및 펌웨어를 효율적으로 실행하는 마이크로프로세서(158)에 의해 관리될 수 있다. 즉, 마이크로프로세서(158)는 코드 기기(미도시)를 운영할 수 있는데, 즉 소프트웨어 코드 기기 또는 펌웨어, 이것은 비휘발성 메모리(174)가 제어될 수 있도록 한다. CPU와 함께 마이크로프로세서(158), 분리된 플래시 ROM 또는 비휘발성 메모리(174)에 패키지되는 플래시 메모리일 수 있는 상기와 같은 코드 기기는 비휘발성 메모리(174)의 물리 블록으로 하여금 어드레스되도록 하며, 정보가 상기 물리 블록으로 기록되거나 또는 이것으로부터 판독 또는 삭제될 수 있도록 한다.

<66> 논리 블록과 관련된 새로운 또는 업데이트된 데이터가 수신될 때, 상기 논리 블록과 관련된 업데이트가 완료되고 다른 논리 블록과 관련된 업데이트가 프로세스되기까지 상기 데이터는 RAM 캐시에 일시적으로 캐시될 수 있다. 도4a는 본 발명의 일 실시예에 상응하여, 논리 블록과 관련된 업데이트들을 캐시하도록 구성된 RAM 캐시, 논리 블록 및 물리 블록의 계통도이다. NAND 플래시 메모리 구성요소와 같은 비휘발성 메모리 구성요소에 위치하고 있는 물리 블록 A(410)은 논리 블록 A(400)에 맵된다. 바람직한 실시예에서, 물리 블록 A(410)은 완전히 채워져 있으며, 따라서 논리 블록 A(400)과 관련된 업데이트된 또는 새로운 콘텐츠를 받아들일 수 없다. 따라서, 논리 블록 A(400)의 논리 페이지(402c)의 콘텐츠가 업데이트될 때, 업데이트된 콘텐츠(406)는 물리 블록(410)에 저장되지 않을 수 있다. 대신, 업데이트된 콘텐츠(406)는 일시적으로 저장되거나 또는 기록된다. 즉, 물리 블록(410)의 현재 콘텐츠가 RAM 캐시(420)로 복사되거나 또는 캐시된 후에 RAM 캐시(420)로 캐시된다.

<67> 업데이트된 콘텐츠(406)를 RAM 캐시(420)로 업데이트하는 단계는 RAM 캐시(420)에 포함되어 있는 더 오래된 콘텐츠에 재기록하는 단계를 포함한다. 예를 들어, 업데이트된 콘텐츠(406)는 이전에 RAM 캐시(420)에 저장되었던 오래된 데이터 콘텐츠를 재 기록할 수 있다. RAM 캐시(420)가 일 실시예에서 전체 메모리 시스템의 어느 일정한 장소에 위치할 수 있지만, RAM 캐시(420)는 비휘발성 메모리 구성요소, 즉 물리 블록 (410)을 효율적으로 제어하는 제어기에 관련될 수 있다.

<68> 어느 일정한 다른 논리 블록이 업데이트되기 전에, 다음의 업데이트가 논리 블록(400)에서 이루어질 때, 다음의 업데이트는 또한 RAM 캐시(420)에 저장될 수 있다. 도4b에 도시되어 있는 것과 같이, 논리 블록 (400)의 페이지(402)에 관련되어 있는 업데이트된 콘텐츠(426)가 존재할 때, 업데이트된 콘텐츠(426)는 RAM 캐시(420)안에 잇는 어느 적절한 위치의 RAM 캐시(420)에 캐시될 수 있다. 업데이트는 논리 블록(400)과 관련되어 있는 현재의 업데이트가 완료되기 전까지 연속적으로 RAM 캐시(420)에 저장될 수 있다. 일 실시예에서, 논리 블록(400)으로의 현재 세트의 업데이트는 서로 다른 논리 블록과 관련된 업데이트가 프로세스되기 전에 프로세스되는 논리 블록(400)의 모든 업데이트일 수 있다.

<69> 논리 블록(400)으로의 현재 세트의 업데이트가 RAM 캐시(420)로 모두 캐시될 때, RAM 캐시(420)로 일시적으로 저장되었던 업데이트, 예를 들어 업데이트된 콘텐츠(406, 426)는 새로운 물리 블록으로 기록되거나 또는 복사될 수 있다. 도4c는 물리 블록과 RAM 캐시, 즉 도4b의 물리 블록(410)과 RAM 캐시(420)의 계통도인데, 본 발명의 일 실시예에 상응하여 새로운 물리 블록으로 기록되는 콘텐츠를 가기고 있다. 일단, 모든 현재의 업데이트들이 RAM 캐시(420)로 저장되면, 새로운 물리 블록(440)은 얻어지고 논리 블록(400)으로 맵된다.

<70> 일반적으로, 논리 블록과 관련되어 있는 가장 최근 또는 업데이트된 콘텐츠는 물리 블록(440)으로 기록된다. 논리 블록(400)과 관련되어 있는 가장 최근의 콘텐츠는 모두 RAM 캐시(420)에 포함된다. 따라서, RAM 캐시(420)의 콘텐츠를 물리 블록(440)으로 복사하는 단계는 물리 블록(44)으로 하여금 물리 블록 도메인의 논리 블록(400)의 버전이 되도록 한다. 물리 블록(440)이 논리 블록(400)과 관련된 모든 현재의 정보를 포함할 때, 물리 블록(410)은 삭제되며, 논리 블록(400)으로부터 분리되거나 맵 해제된다. 결국, 논리 블록(400)은 전형적으로 소정의 시간에 단지 단일 물리 블록으로 맵된다. 따라서, 물리 블록(440)이 논리 블록(400)과 관련되는 시간과 동시에 물리 블록(410)이 논리 블록(400)과 분리되며, 여유 블록 폴로 반환될 수 있기 때문에, 시스템에서 물리 블록의 사용은 효율적으로 경제적일 수 있다.

<71> 도5를 참고로, 본 발명의 일 실시예에 상응하여, RAM 캐시를 통해 물리 블록 도메인의 페이지 콘텐츠를 업데이트하는 일 방법과 관련된 단계들이 이하에서 설명된다. RAM 캐시를 사용하여 일시적으로 페이지 업데이트를 저장하는 프로세스(500)는 단계(504)에서 시작되는데, 호스트는 논리 블록 A'를 프로세스하기 시작한다. 비휘발성 메모리와 인터페이스하는 시스템, 예를 들어 호스트 또는 비휘발성 메모리가 내장되어 있는 시스템은 논리 블록 'A'에 포함되어 있는 페이지와 관련되어 있는 업데이트를 프로세스할 때 논리 블록 A를 프로세스하기 시작한다.

<72> 논리 블록 A와 관련되어 있는 물리 블록 A의 가용 공간이 존재하지 않을 때, 물리 블록 A는 효율적으로 업데이트를 수용할 수 없다. 따라서, RAM 캐시는 상기 업데이트가 새로운 물리 블록에 저장되기 전에 업데이트를 캐시하는데 사용된다. 단계(504)로부터, 프로세스 흐름은 새로운 물리 블록 B가 얻어진 단계(505)로 진행된다. 새로운 물리 블록 B가 예를 들어 여유 블록의 세트로부터, 논리 블록 A로 맵되도록 또는 논리 블록 A와 관련되도록 얻어진다. 물리 블록 A의 모든 콘텐츠는 단계(506)에서 RAM 캐시로 기록되거나 도는 캐시된다. 단계(508)에서, 논리 블록 A에 포함되어 있는 페이지로의 업데이트는 예를 들어 제어기에 의해 RAM 캐시로 저장된다. 일반적으로, 업데이트는 업데이트된 데이터 또는 새로운 데이터를 포함할 수 있으며, 상기 업데이트는 물리 블록 A와 관련되어 있는 데이터로 하여금 RAM 캐시로 재기록되도록 한다. 도4a-c와 관련하여 상기 설명된 것과 같이, RAM 캐시는 RAM 또는 일시적으로 업데이트된 데이터를 사용하는데 유용한 메모리이다. 일단 상기 논리 페이지로의 업데이트가 상기 RAM 캐시로 저장되면, 상기 호스트가 다른 논리 블록 예를 들어 논리 블록 B를 프로세스하기 시작할 준비가 되었는지에 대해 단계(512)에서 결정이 이루어진다. 달리 말하면, 다른 블록이 프로세스되기 전에 RAM 캐시에 저장되어야하는 논리 블록과 관련되어 있는 추가의 업데이트가 있는지에 대해 결정한다.

<73> 만약 단계(512)에서의 결정이 블록 B의 처리를 시작한 시간이 아니라는 것이면, 논리 블록 A와 관련되어 잇는 추가의 업데이트가 있다는 것을 지시한다. 따라서, 프로세스 흐름은 논리 블록 A와 관련되어 있는 업데이트가 RAM 캐시로 저장되는 단계(508)로 반환된다. 대안적으로, 만약 다른 논리 블록을 프로세스하여야 할 시간이 아니라고 결정되면, RAM 캐시의 콘텐츠는 단계(518)에서 물리 블록 B에 기록되며, 따라서 논리 블록 A와 관련된 가장 최근의 콘텐츠는 물리 블록 B로 저장되거나 또는 기록된다.

<74> 일단 정보가 물리 블록 B에 저장되면, 물리 블록 A는 단계 (524)에서 삭제된다. RAM 캐시에 저장된 정보가 재기록될 수 있기 때문에, RAM 캐시의 콘텐츠가 물리 블록 B에 통합된 후에 상기 RAM 캐시를 반드시 삭제하여야 하는 것은 아니라는 것을 이해할 수 있을 것이다. 전형적으로, 또 다른 물리 블록의 콘텐츠가 상기 RAM 캐시에 저장될 때, 상기 RAM 캐시의 이전 콘텐츠는 재기록된다. 일 실시예에서, 일단 RAM 캐시의 콘텐츠가 물리 블록 B에 통합되면, RAM 캐시는 삭제될 수 있다.

<75> 물리 블록 A가 단계(524)에서 삭제된 후에, 물리 블록 A는 논리 블록 A로부터 단계(528)에서 분리된다. 논리 블록 A로부터 물리 블록 A의 분리는 최초 논리 블록 A가 물리 블록 B에 관련되어 있다는 것을 지시하는 업데이트 맵핑을 포함할 수 있다. 논리 블록 A로부터 물리 블록 A를 분리할 때, 논리 블록과 관련되어 있는 페이지 업데이트를 일시적으로 저장하는 RAM 캐시를 사용하는 프로세스가 완료된다.

<76> 상기 업데이트를 물리 블록에 기록하기 전에 논리 블록과 관련되어 있는 업데이트를 일시적으로 저장하는 RAM 캐시의 사용은 비록 효율적이지만, 메모리 시스템은 RAM 캐시로 사용되는 가용 RAM을 항상 가지고 있는 것은 아니다. 전형적으로, RAM 캐시는 블록의 모든 콘텐츠를 수용할 수 있을 정도의 사이즈이다. 따라서, RAM 캐시의 사이즈는 시스템에 있는 물리 블록의 사이즈에 의존하여 변화될 수 있다. 예를 들어, 만약 시스템 안에 있는 물리 블록이 거의 32페이지를 가지고 있으면, 상기 RAM 캐시는 블록의 32페이지의 콘텐츠를 수용하기에 충분할 정조의 사이즈일 수 있다. 물리 블록을 수용하기에 적당한 사이즈의 RAM 캐시와 같은 소프트웨어 요구가 비가용이면, 실제 물리 블록은 상기 정보가 논리 블록과 관련되어 있는 물리 블록에 기록되기 전에 일시적으로 정보를 유지하는데 사용될 수 있다. 달리 말하면, RAM 캐시인 "캐싱 블록"을 구현하기보다는, 물리 블록인 캐싱 블록이 대신 구현될 수 있다.

<77> 도6a는 본 발명의 일 실시예에 상응하여, 논리 블록, 물리 블록 및 논리 블록과 관련되어 있는 업데이트를 캐시하도록 구성된 캐싱 블록, 즉 물리 블록 캐시의 계통도이다. 비휘발성 메모리 구성요소인 물리 블록 A(610)은 논리 블록 A(600)로 맵된다. 물리 블록 A(610)가 완전히 채워지고 논리 블록 A(600)과 관련된 업데이트된 또는 새로운 콘텐츠를 수용할 수 없을 때, 물리 블록 캐시(612)는 상기 업데이트된 또는 새로운 콘텐츠가 일시적으로 캐시되는데 사용될 수 있다.

<78> 도시되어 있는 것과 같이, 논리 블록(600)의 논리 페이지(602c)는 업데이트된 콘텐츠(606)를 포함한다. 업데이트된 콘텐츠(606)가 논리 페이지(602c)와 관련되어 있다는 것이 식별되도록, 업데이트된 콘텐츠(606)가 논리 페이지(602c)와 관련되어 있다는 지시가 물리 블록 캐시(612)에 저장되거나 가용된다면, 업데이트된 콘텐츠(606)는 물리 블록 캐시(612) 안의 어느 일정한 장소에 저장될 수 있다. 예로써, 업데이트된 콘텐츠(606)는 도시되어 있는 것과 같이, 물리 블록 캐시(612) 안에 있는 제1 가용 페이지로 캐시될 수 있다. 대안적으로, 일 실시예에서, 업데이트된 콘텐츠(606)는 논리 페이지(602c)에 상응하는 위치의 물리 블록 캐시(612)에 기록되거나 또는 캐시될 수 있다.

<79> 예를 들어, 업데이트된 콘텐츠(616)가 논리 페이지(602c)에 관련될 때 효율적으로 이루어지는 다음 업데이트가 존재할 때, 업데이트된 콘텐츠(616)는 또한 물리 블록 캐시(612)에 캐시될 수 있다. 일단, 논리 블록 A(600)과 관련되어 있는 모든 업데이트가 일시적으로 완료될 때, 예를 들어 다른 논리 블록(미도시)을 프로세스할 때, 논리 블록 A(600)과 관련되어 있는, 업데이트(606, 616)을 포함하는 가장 최근의 데이터는 도6b와 관련하여 설명되는 것과 같이 새로운 물리 블록에 기록될 수 있다. 일정한 실시예에서, 논리 블록 A(600)과 관련되어 잇는 "가장 최근의 데이터"는 단지 업데이트만을 포함할 수 있는 반면, 도시되어 있는 실시예를 포함하는 다른 실시예에서, 논리 블록 A(600)고 관련되어 있는 가장 최근 데이터는 업데이트(6060, 616) 및 업데이트(606, 616)에 의해 빼앗기지 않은 물리 블록 A(610)의 일정한 콘텐츠를 포함한다. 물리 블록(612)가 완전히 채워질 때, 또는 다른 논리 블록을 프로세스하여야할 시간이 아닐 때, 통합 작동 또는 유사한 작동이 논리 블록 A(600)와 관련되 가장 최근의 콘텐츠를 새로운 물리 블록에 제공하도록 발생할 수 있다는 것을 이해할 수 있을 것이다. 상기 새로운 물리 블록은 "이전" 물리 블록으로 되며, 논리 블록 A(600)과 관련된 업데이트는 새로운 물리 블록(612)에 캐시될 수 있다.

<80> 도6b에 도시되어 있는 것과 같이, 다른 논리 블록이 프로세스된다고 가정하는 것은 논리 블록 A(600)로 맵되는 새로운 물리 블록 B(620)을 얻을 수 있다는 것이다. 새로운 물리 블록 B(620)은 요구되는 방식으로 또는 일 실시예에서 논리 블록 A(600)과 관련되어 있는 캐싱 프로세스가 처음으로 시작될 때 얻어질 수 있다. 일반적으로, 가장 최근 또는 업데이트된, 논리 블록 A(600)와 관련되어 있는 콘텐츠는 물리 블록B(620)에 기록된다. 종종, 논리 블록 A(600)과 관련되어 있는 가장 최근의 콘텐츠는 물리 블록 A(610) 및 물리 블록 캐시(612) 안에 포함될 수 있다. 특히, 논리 블록 A(600)의 어느 일정한 업데이트된 콘탠츠는 물리 블록 캐시(612)에 포함되는 반면, 현재의 업데이트 세트의 결과로 업데이트되지 않은 콘텐츠 또는 더 이전의 콘텐츠가 물리 블록 A(610)에 포함된다. 논리 블록 A(600)의 어느 일정한 업데이트된 콘텐츠는 논리 블록 A(600)안에 포함되어 있는 가장 최근 업데이트된 콘텐츠에 의해 바뀌어지며, 더 새롭게 업데이트된 콘텐츠는 효율적으로 업데이트된 콘텐츠(606)와 바뀐다는 것을 이해할 수 있을 것이다.

<81> 물리 블록 A(610)과 물리 블록 캐시(612)의 콘텐츠는 논리 블록(620)의 페이지가 물리 블록 캐시(612)로부터 얻어진 업데이트된 콘텐츠(606, 616)와 물리 블록 캐시(612)의 콘텐츠에 의해 바뀌어지지 않은 물리 블록 A(610)의 현재 콘텐츠인 물리 블록 A(610)의 콘텐츠를 포함하도록 효율적으로 결합된다. 일정한 경우에, 물리 블록 A(610)의 콘텐츠는 물리 블록의 어떠한 콘텐츠와 물리 블록 B(620)에 포함되지 않도록 물리 블록 캐시(612)의 콘텐츠로 바뀌어질 수 있다. 일단 물리 블록 B(620)가 물리 블록 도메인의 논리 블록 A(600)의 버전이고, 따라서 논리 블록 A(600)과 관련된 현재의 모든 정보를 포함한다면, 물리 블록 A(610)은 삭제될 것이다.

<82> 도7는 본 발명의 일 실시예에 상응하여, 물리 블록 캐시의 사용을 통해 물리 블록 도메인의 페이지 콘텐츠를 업데이트하는 한 방법에 관련된 단계들을 설명하는 프로세스 흐름도이다. 논리 페이지 업데이트를 일시적으로 저장하기 위해 블록 캐시를 사용하는 프로세스(660)는 단계(670)에서 시작되는데, 호스트는 논리 블록 A를 프로세스하기 시작한다. 달리 말하면, 비휘발성 메모리가 인터페이싱하는 시스템 또는 비휘발성 메모리가 내장되어 있는 시스템은 논리 블록 A의 콘텐츠의 업데이트를 프로세스하기 시작한다.

<83> 상기 호스트가 논리 블록 A를 프로세스하기 시작한 후에, 논리 블록 A에 맵되는 새로운 물리 블록 B는 단계(672)에서 얻어진다. 논리 블록 A에 포함되어 있는 페이지의 업데이트는 단계(674)에서 블록 캐시로 저장되는데, 예를 들어 제어기에 의해 그러하다. 도6a와 6b에 관련하여 상기에서 설명한 것과 같이, 논리 블록 A로 맵되는 물리 블록 A가 페이지 업데이트를 수용할 수 없을 때, 상기 페이지 업데이트는 캐시에 저장될 수 있다. 일단 논리 블록 A의 페이지 업데이트가 블록 캐시에 저장된 후에, 상기 호스트가 다른 논리 블록, 예를 들어 논리 블록 B를 프로세스하기 시작할 준비가 되었는지에 대한 결정이 단계(678)에서 이루어진다. 즉, 프로세스되어야 하는 논리 블록 A와 관련되어 있는 추가의 업데이트가 있는지가 결정된다.

<84> 만약 단계(678)에서 논리 블록 B를 프로세스 시작할 때가 아니라고 결정되면, 논리 블록 A와 관련된 추가의 업데이트가 있다는 것을 지시한다. 따라서, 프로세스 흐름은 단계(674)로 반환되는데, 여기서 논리 블록 A와 관련된 업데이트는 블록 캐시로 저장된다. 한편, 만약 논리 블록 B를 프로세스 시작할 때가 아니라고 결정되면, 물리 블록 A와 블록 캐시는 단계(686)에서 물리 블록 B에 결합하며, 논리 블록 A와 관련된 가장 최근의 콘텐츠는 물리 블록 B에 기록된다. 즉, 물리 블록 A의 콘텐츠와 블록 캐시의 콘텐츠는 효율적으로 통합되어 물리 블록 B에 저장되며, 물리 블록 B는 논리 블록 A의 페이지와 관련된 현재의 콘텐츠를 포함한다. 일정한 경우에, 블록 캐시의 콘텐츠는 효율적으로 물리 블록 A의 콘텐츠를 바꾸며, 물리 블록 A의 어떠한 콘텐츠도 복사되거나 또는 물리 블록 B로 저장되지 않는다.

<85> 물리 블록 A는 논리 블록 A와 관련된 현재의 콘텐츠가 물리 블록 B에 기록된 후에 단계(690)에서 삭제된다. 단계(694)에서, 상기 블록 캐시는 삭제된다. 일단 상기 블록 캐시가 삭제되면, 물리 블록 A는 단계(698)에서 논리 블록 A로부터 분리된다. 물리 블록 A를 논리 블록 A로부터 분리하는 단계는 논리 블록 A가 물리 블록 B와 관련되어 있다는 것을 지시하는 맵핑을 업데이트하는 단계를 포함한다. 물리 블록 A가 논리 블록 A로부터 분리된 후, 논리 블록과 관련되어 있는 페이지 업데이트를 일시적으로 저장하기 위해 블록 캐시를 사용하는 프로세스는 완료된다.

일반적으로, RAM 캐시 또는 블록 캐시가 활용되도록 하는 것과 관련된 기능은 프로그램 코드 기기 또는 펌웨어와 같은 프로그램으로 호스트 시스템에 제공된다. 본 발명의 일 실시예에 상응하여, 호스트 시스템으로 제공되는 소프트웨어 또는 펌웨어와 관련되어 있는 적절한 시스템 구조의 한 실시예가 도8에 도시되어 있다. 시스템(700)은 일반적으로 애플리케이션 모듈(704), 시스템 관리자 모듈(708), 데이터 관리자 모듈(712), 데이터 무결성 관리자(716) 및 인터페이스 모듈(720) 및 기기 관리자를 포함하는, 그러나 이에 한정되지 않는 여러 모듈을 포함한다. 일반적으로, 시스템 구조(700)는 예를 들어 도3a의 프로세서(108)와 같은 프로세서에 의해 액세스될 수 있는 소프트웨어 코드 기기 또는 펌웨어를 사용하여 구현될 수 있다.

<86> 일반적으로, 애플리케이션 인터페이스 모듈(704)는 호스트, 작동 시스템 또는 사용자와 직접적으로 통신하도록 구성되어 있다. 애플리케이션 인터페이스 모듈(704)은 또한 시스템 관리자 모듈(708) 및 데이터 관리자 모듈(712)과 통신할 수 있다. 사용자가 플래시 메모리를 판독하고, 기록하고 포맷하기를 원할 때, 사용자는 요구를 상기 작동 시스템으로 전송하며, 상기 요구는 상기 애플리케이션 인터페이스 모듈(704)로 전송된다. 애플리케이션 모듈(704)은 상기 요구를 상기 요구에 따라 시스템 관리자 모듈(708) 또는 데이터 관리자 모듈(712)로 전달한다.

<87> 시스템 관리자 모듈(708)은 시스템 초기화 서브모듈(724), 삭제 카운트 블록 관리 서브모듈(726) 및 전력 관리 블록 서브모듈(730)을 포함한다. 시스템 초기화 서브모듈(724)은 일반적으로 초기화 요구가 프로세스되고 삭제 카운트 블록 관리 서브모듈(726)과 통신할 수 있도록 구성된다.

<88> 애플리케이션 인터페이스 모듈(704)과 통신하는 것에 추가하여, 시스템 관리자 모듈(708)은 또한 기기 관리자 및 인터페이스 모듈(70)뿐만 아니라 데이터 관리자 모듈(712)과 통신할 수 있다. 시스템 관리자 모듈(708)과 애플리케이션 인터페이스 모듈(704) 모두와 통신하는 데이터 관리자 모듈(712)은 페이지 또는 블록 맵핑을 제공하는 기능을 포함한다. 데이터 관리자 모듈(712)은 또한 작동 시스템과 관련된 기능과 파일 시스템 인터페이스 계층을 포함할 수 있다.

<89> 시스템 관리자 모듈(708), 데이터 관리자(712) 및 데이터 무결성 관리자(716)와 통신하는 기기 관리자와 인터페이스 모듈(720)는 전형적으로 플래시 메모리 인터페이스를 제공하며, I/O 인터페이스와 같은 하드웨어 추상화와 관련된 기능을 포함한다. 데이터 무결성 관리자 모듈(716)은 다른 기능들 중에서 ECC 핸들링을 제공한다.

<90> 본 발명의 일 실시예에서, 캐싱이 페이지 방식으로 발생되도록 블록 캐싱을 수행하는 대신에, 블록 캐싱이 그룹으로 수행될 수 있다. 미국 특허 출원 제 10/281,855에 설명되어 있는 것과 같이, 그룹은 블록 안에 일정한 수의 페이지를 포함한다. 예를 들어, 32페이지를 포함하는 블록은 상기 페이지들이 8개의 페이지를 각각 포함하고 있는 4개의 그룹으로 그룹짓도록 구성될 수 있다. 일반적으로, 블록의 그룹 수와 그룹의 페이지 수는 크게 변화할 수 있다.

<91> 그룹핑 구조가 사용될 때, 업데이트된 데이터는 그룹의 부분으로 물리 블록에 기록될 수 있다. 도9a는 본 발명의 일 실시예에 상응하여, 그룹으로 분리되는 새로운 물리 블록, 논리 블록 및 물리 블록의 계통도이다. 논리 블록 A(900)은 논리 그룹(904)으로 분리되는데, 각각은 다수의 논리 페이지(908)을 포함하고 있다. 비록 논리 블록 A(900)는 각각 4개의 페이지들(908)를 포함하고 있는 4개의 논리 그룹(904)를 포함하도록 도시되어 있지만, 논리 블록 A(900)에 포함되어 있는 그룹(904)의 수와 페이지(908)의 수는 크게 변화될 수 있다. 특정 페이지(908)가 속해 있는 그룹(904)는 전형적으로 각 페이지(908)와 관련되어 있는 여분의 영역 또는 오버헤드에서 식별된다.

<92> 물리 블록B(9122)는 물리 블록 B(912)가 논리 블록 A(900)와 관련되어 있는 콘텐츠 또는 데이터를 포함하도록 논리 블록 A(900)와 관련되어 있다. 도시되어 있는 것과 같이, 물리 페이지를 포함하고 있는, 물리 블록 B(912)의 제1 물리 그룹(916a)은 논리 그룹(904a)과 관련되어 있는 데이터(920a)을 포함한다. 유사하게, 제1 물리 그룹(916b)은 논리 그룹(904b)과 관련되어 있는 데이터(920b)을 포함하며, 제3 물리 그룹(916c)은 논리 그룹(904c)과 관련되어 있는 데이터(920c)을 포함하며, 제4 물리 그룹(916d)은 논리 그룹(904d)과 관련되어 있는

데이터(920d)을 포함한다.

<93> 업데이트(924)이 예를 들어 호스트로부터 수신될 때(이것은 논리 그룹(904b)과 관련되어 있는 데이터가 업데이트된다는 것을 지시한다), 업데이트(924)은 효율적으로 데이터(920b)와 통합되며, 새로운 물리 블록 B1(928)의 제1 가용 그룹(923a)으로 저장된다. 통합된 논리 그룹(904b)의 각 페이지와 관련되어 있는 가장 최근의 콘텐츠가 그룹(93a)에 저장되도록 한다. 업데이트된 그룹과 관련된 데이터와 최초의 데이터의 통합은 도10을 참고하여 이하에서 설명될 것이다. 도9b에 도시되어 있는 것과 같이, 논리 블록 A(900)와 관련된 업데이트 프로세스가 완료될 때, 물리 블록B1(928)은 논리 블록A(900)와 관련된 물리블록, 즉 물리 블록 B1(928)은 물리 블록 B(912)가 논리 블록 A(900)으로부터 분리되는 동안에 논리 블록 A(900)와 관련되게 된다.

<94> 물리 블록 B(912)가 논리 블록 A(900)으로부터 분리되기 전, 물리 블록 B(912)의 콘텐츠는 물리 블록 B1(928)로 통합되지 않는다. 예를 들어, 제1 논리 그룹(904a)과 관련되어 있는 데이터(920b)는 논리 그룹(904a)에 관련되어 있는 데이터(920c)가 제3 물리 그룹(932c)에 제공되는 동안 물리 블록 B(928)의 제2 물리 그룹(932b)에 복사되거나 또는 제공된다. 유사하게, 제4 논리 그룹(904d)과 관련되어 있는 데이터(920d)는 제4 물리 그룹(932d)으로 제공될 수 있다. 결국, 논리 블록 A(900)와 관련되어 있는 가장 최근의 콘텐츠는 물리 블록 B1(928)에 존재한다.

<95> 물리 블록 B1(928)이 논리 블록 A(900)와 관련된 물리 블록이 된 후, 추가의 업데이트는 논리 블록 A(900)에서 이루어 질 수 있다. 도9c를 참고하여, 이전에 업데이트되었던 그리고 그룹(904)으로 분리되는 논리 블록 A(900)로의 추가 업데이트를 프로세스하는 것은 본 발명의 일 실시예에 상응하여 설명될 것이다. 논리 그룹(904b)과 관련된 새로운 업데이트(940)이 프로세스될 때, 물리 블록 B(912)는 더 이상 논리 블록과 관련되어 있는 가장 최근의 콘텐츠를 포함하지 않으며, 물리 블록 B1(928)은 효율적으로 완전히 채워지므로, 새로운 물리 블록 B2(944)는 업데이트된(940)을 저장하도록 얻어진다. 특히, 논리 그룹(904b)에 관련되어 있는, 물리 그룹(932a)에 저장되는 데이터(936)는 업데이트(940)과 통합되며, 업데이트된 데이터(948)와 같이 물리 블록 B2(944)의 제1 가용 그룹으로 저장된다. 업데이트된 데이터(948)는 논리 그룹(904b)안의 페이지와 관련되어 있는 가장 최근의 콘텐츠를 포함한다.

<96> 일단 업데이트 프로세스가 완료된 후, 논리 블록 A(900)와 관련된 가장 최근의 데이터는 물리 블록 B2(944)에 제공된다. 예로서, 제1 논리 그룹(904a)과 관련된 데이터(920a)은 물리 블록 B2(944)의 제2 물리 그룹(946b)에 제동될 수 있으며, 제3 논리 그룹(904c)과 관련된 데이터(920c)은 제3 물리 그룹(946c)에 제동될 수 있으며, 제4 논리 그룹(904d)과 관련된 데이터(920d)은 제4 물리 그룹(946d)에 제동될 수 있다. 물리 블록 B2(944)는 논리 블록 A(900)와 관련된 가장 최근의 콘텐츠를 포함할 때, 물리 블록 B1(928)는 논리 블록 A(900)으로부터 분리되면서 물리 블록 B2(944)는 논리 블록 A(900)에 관련될 수 있다.

<97> 그룹으로 분리되는 논리 블록이 초기에 콘텐츠를 가지고 있지 않다면, 논리 블록과 관련된 새로운 또는 업데이트된 데이터가 예를 들어 호스트로부터 수신될 때, 업데이트된 또는 새로운 데이터는 물리 블록에 저장되어 있는 다른 데이터가 효율적으로 상기 데이터 중 일부를 바꾸더라도 물리 블록으로 저장될 수 있다. 도9d는 본 발명의 일 실시예에 상응하게 그룹으로 분리되는 논리 블록, 상기 논리 블록과 관련된 캐싱된 데이터를 포함하고 있는 물리 블록 및 상기 논리 블록에 관련된 가장 최근 데이터를 포함하고 있는 새로운 물리 블록의 계통도이다. 논리 블록 A(950)가 초기에 콘텐츠를 포함하지 않을 때, 콘텐츠가 논리 블록 A(950)와 관련될 때, 상기 콘텐츠는 논리 블록 A(950)와 관련되어 있는 물리 블록 B(956)에 기록될 수 있다. 도시되어 있는 것과 같이, 논리 블록 A(950)의 제2 논리 그룹(952d)과 관련되어 있는 새로운 데이터(960a)이 제공될 때, 데이터(960a)은 물리 블록 B(956)의 제1 가용 물리 그룹(958a)으로 저장될 수 있다.

<98> 제2 논리 그룹(952b)과 관련된 데이터가 바뀌어질 때, 업데이트된 데이터(960a')은 다음 가용 물리 그룹(958)으로, 예를 들어 제2 물리 그룹(958b)으로 저장될 수 있다. 데이터(960a)이 저장된 후, 만약 제3논리 그룹(952c)과 관련되어 있는 새로운 데이터(960b)이 수신되면, 데이터(960b)은 다음 가용 물리 그룹(958), 즉 제3 물리 그룹(958c)에 저장된다. 마지막으로, 만약 제2 논리 그룹(952b)과 관련되어 있는 업데이트된 데이터(960a')이 수신되면, 업데이트된 데이터(960a')은 설명된 실시예에서, 물리 블록 B(956)의 마지막 물리 가용 그룹(958)인 제4 물리 그룹(958d)에 저장된다.

<99> 일단 물리 블록 B(956)가 완전히 채워지면, 즉 물리 블록 B(956)에 있는 모든 그룹들이 데이터(960)를 포함한다면, 논리 블록 A(950)와 관련되어 있는 새로운 또는 업데이트된 데이터가 수신되며, 새로운 물리 블록은 일반적으로 사용을 위해 얻어진다. 예로써, 만약 논리 그룹(952d)의 새로운 데이터(966)가 수신된다면, 새로운 물리 블록 B1(962)는 데이터(966)를 수용하기 위해 얻어진다. 데이터(966)는 물리 블록 B1(962)의 제1 가용 그룹

(964a)으로 저장된다. 데이터(966)이 저장된 후, 물리 블록 B(956)에 포함되어 있는 데이터(960)는 물리 블록 B2(962)로 저장된다.

<100> 물리 블록 B(956)에 포함되어 있는 데이터(960)를 물리 블록 B2(962)에 저장하기 위해, "하부 먼저"의 방식으로, 즉 그룹(958d)에서 시작하여 그룹(958a)으로 끝나는 방식으로 물리 블록 B(956)로부터 판독될 수 있다. 만약 데이터가 논리 그룹(952b)에 관련되어 있는 데이터(960)를 포함하고 있는 물리 그룹(958a, 958b, 958d) 안에 있는 동일한 페이지 오프셋으로 존재하면, 물리 그룹(958d)의 페이지 오프셋에 포함되어 있는 데이터는 가장 최근 데이터로 평가된다. 일반적으로, 물리 그룹의 번호가 더 상위일수록, 동일한 그룹의 더 새로운 데이터이다. 일단 논리 그룹(952b)에 관련되어 있는 가장 최근 데이터가 식별되면, 또는 일단 데이터(960a, 960a', 960a'')가 적절하게 통합된다면, 논리 그룹(952b)에 관련되어 있는 통합된, 그리고 업데이트된 데이터(968)는 물리 블록 B1(962)안에 있는 그룹(964b)으로 저장된다. 물리 블록 B(956) 안에 있는 단지 하나의 그룹(958c)이 논리 그룹(952d)과 관련되어 있는 데이터(960b)를 포함하기 때문에, 데이터(960b)는 물리 블록B1(962)의 제3 그룹(964c)에 제공될 수 있다. 어떠한 데이터도 그룹(964d)에 저장되지 않기 때문에, 그룹(964d)은 효율적으로 새로운 논리 그룹, 예를 들어 논리 그룹(952a)과 관련되어 있는 데이터를 수용할 수 있다.

이전에 언급한 것과 같이, 논리 그룹에 대한 업데이트된 데이터가 수신될 때, 만약 더 이전 데이터가 상기 그룹을 위해 존재하면, 더 이전 데이터는 적절하다면 업데이트된 데이터에 통합될 수 있다. 일 실시예에서, 업데이트된 데이터가 상기 그룹 안에 있는 각 페이지에 대한 업데이트된 데이터를 포함할 때, 상기 업데이트된 데이터는 효율적으로 모든 이전 데이터를 재기록한다. 그러나, 상기 업데이트된 데이터가 상기 그룹 안에 있는 일정한 페이지만에 대한 업데이트를 포함할 때, 통합 프로세스가 발생된다. 도10을 참고로, 그룹의 최초 데이터를 그룹의 새로운 데이터에 통합하는 한 방법은 본 발명의 일 실시예에 상응하여 설명될 것이다. 그룹 X(1000)은 8개의 페이지들(1002)을 포함할 수 있다. 일정한 페이지(1002)은 데이터를 포함하는 반면, 다른 페이지(1002)은 비어있을 수 있다. 도시되어 있는 것과 같이, 페이지(1002a, 1002d, 1002e)은 최초 데이터를 포함한다.

<101> 새로운 데이터 그룹X(1000)이 호스트로부터 업데이트의 형태로 수신될 때, 예를 들어, 새로운 데이터는 그룹 업데이트(1006)의 페이지(1008)에 제공된다. 페이지(1008a-d)와 페이지(1008g)는 그룹 X(1000)에 포함되어 있는 데이터와 통합되는 새로운 또는 업데이트된 데이터를 포함한다. 통합 프로세스는 전형적으로 그룹 X(1000)와 관련된 가장 최근 데이터를 식별하는 단계를 포함한다. 새로운 데이터는 효율적으로 최초 데이터를 재기록하기 때문에, 상기 새로운 데이터는 그룹 X(1000)와 관련된 가장 최근의 데이터로 고려된다. 한편, 최초 데이터를 효율적으로 재기록하는 새로운 데이터가 존재하지 않을 때, 상기 최초 데이터는 가장 최근 데이터로 고려된다.

<102> 페이지(1008a)로부터의 새로운 데이터는 효율적으로 제1 페이지(1002)에 저장되어 있는 최초 데이터를 바꾸기 때문에, 그룹 X(1000)에 대한 최초 데이터와 그룹 업데이트(1006)에 제공된 새로운 데이터를 통합한 결과를 포함하고 있는 통합된 데이터(1010)는 그룹 X(1000)의 제1 페이지(1002)에 상응하는 제1 페이지(1012a)가 페이지(1008a)의 새로운 데이터를 포함하도록 된다. 최초에 페이지(1002b, 1002c)에 포함되어 있는 데이터는 없기 때문에, 페이지(1008b, 1008c)에 의해 저장되는 새로운 데이터는 페이지(1012b와 1012c)에 각각 기록된다. 페이지(1008d)에 포함되어 있는 데이터는 페이지(1002d)에 포함되어 있는 데이터보다 더 최근의 것이기 때문에, 페이지(1012d)는 페이지(1008d)로부터의 새로운 데이터를 포함하고 있다. 그룹 업데이트(1006)은 페이지(1008e)의 업데이트된 콘텐츠를 포함하고 있지 않기 때문에 페이지(1002e)로부터의 최초 데이터를 포함한다. 마지막으로, 페이지(1012g)는 페이지(1008h)에 포함되어 있는 새로운 데이터를 포함한다. 일단 통합된 데이터(1010)가 완료되면, 통합된 데이터(1010)는 효율적으로 그룹 X(1000)가 된다.

<103> 일반적으로, 그룹으로의 업데이트가 특정 페이지와 관련되어 있는 데이터를 포함할 때, 상기 데이터는 특정 페이지와 관련되어 있는 데이터이거나 또는 상기 데이터는 상기 특정 페이지와 관련되어 있는 최초 데이터를 재기록하기 때문에, 상기 데이터는 상기 그룹에 대한 결과 통합 데이터 안에 있는 상응하는 페이지에 포함되어 있다. 대안적으로, 소정 페이지의 최초 데이터가 업데이트에 의해 상기 페이지를 포함하는 상기 그룹으로 재기록되지 않을 때, 상기 최초 데이터는 상기 그룹을 위한 통합된 결과 데이터 안에 있는 상응하는 페이지에 포함된다.

<104> 도11a와 11b는 본 발명의 일 실시예에 상응하는, 그룹으로 분리되는 논리 블록에 관련되어 있는 상기 콘텐츠를 업데이트하는 한 방법에 관련되어 있는 단계를 설명하는 흐름도이다. 콘텐츠를 업데이트하는 프로세스(1100)가 단계(1102)에서 시작되는데, 여기서 물리 블록 A와 관련되어 있는 논리 블록 A와 관련되는 그룹 X와 관련되어 있는 콘텐츠 예를 들어 호스트에 의해 업데이트된다. 전형적으로, 논리 그룹X는 2개 이상의 페이지를 포함한다. 일단 논리 블록 A의 그룹X와 관련되어 있는 콘텐츠가 업데이트되면, 새로운 물리 블록B1이 단계

(1104)에서 논리 블록 A와 관련되어 얻어진다.

<105> 단계(1106)에서, 논리 블록 X와 관련되어 있는, 전형적으로 물리 블록 B 안에 있는 그룹에 저장되는 이전 콘텐츠는 논리 그룹 X에 대한 새로운 또는 업데이트된 콘텐츠와 통합되며, 물리 블록 B1에 저장된다. 상기 통합된 콘텐츠가 물리 블록 B1에 저장된 후, 저장되어야 하는 논리 블록 A에 대한 추가의 새로운 또는 업데이트된 콘텐츠가 존재하는지에 대한 결정이 이루어진다. 만약 상기 결정이 저장되어야 하는 추가의 새로운 또는 업데이트된 콘텐츠가 있다는 것이면, 프로세스 흐름은 단계(1105)로 반환되는데, 여기서 논리 그룹 X와 관련되어 있는, 논리 블록 A 안에 있는 다른 논리 그룹일 수 있는 이전 콘텐츠 새로운 또는 업데이트된 콘텐츠와 통합된다. 달리 말하면, 새로운 또는 업데이트된 콘텐츠는 논리 블록 A 안에 있는 어느 일정한 그룹과 관련되어 있으며, 따라서 상기 통합은 상기 새로운 또는 업데이트된 콘텐츠와 적절한 논리 그룹 사이에서 발생될 것이다.

<106> 대안적으로, 만약 단계(1108)에서의 결정이 논리 블록 A에 대한 저장되어야 하는 추가의 새로운 또는 업데이트된 콘텐츠가 존재하지 않는다는 것이면, 단계(1110)에서 나머지 콘텐츠 즉, 물리 블록 B의 물리 블록 B1로 통합되지 않은 콘텐츠를 저장하기 위한 물리 블록 B1의 충분한 여유가 있는지에 대한 결정이 이루어진다. 즉, 논리 블록 A와 관련되어 있는 현재의 콘텐츠의 완전하게 채워짐을 물리 블록 B에 저장될 수 있는지를 결정한다. 물리 블록 B의 나머지 콘텐츠를 물리 블록 B1에 저장할 수 있는 물리 블록 B1의 충분한 가용 그룹이 있다고 결정이 되면, 단계(1120)에서 상기 물리 블록 B로부터의 나머지 콘텐츠는 물리 블록 B1로 복사된다.

<107> 물리 블록 B로부터의 잔여 콘텐츠가 물리 블록 B1에 복사된 후, 물리 블록 B1은 효율적으로 가장 최근의 콘텐츠 또는 논리 블록 A와 관련되어 있는 현재의 콘텐츠를 포함한다. 따라서, 물리 블록 B는 단계(1122)에서 삭제되며, 단계(1124)에서 논리 블록 A로부터 분리된다. 일단 물리 블록 B가 논리 블록 A로부터 분리되면, 논리 블록과 관련되어 있는 콘텐츠를 업데이트하는 프로세스가 완료된다.

<108> 단계(111)로 반환되어, 만약 물리 블록 B의 잔여 콘텐츠를 저장하는 물리 블록 B1의 충분한 가용 그룹이 존재하지 않는다고 결정이 되면, 새로운 블록이 논리 블록 A와 관련된 가장 최근의 콘텐츠를 저장하기 위해 얻어진다는 것을 지시한다. 따라서, 프로세스 흐름은 단계(1112)로 이동하는데, 여기서 새로운 물리 블록 B2는 논리 블록 A와 관련되어 할당된다. 일단 새로운 블록 B2가 얻어지면, 물리 블록 B와 물리 블록 B1의 콘텐츠는 단계(114)에서 물리 블록 B2로 통합되어, 논리 블록 A와 관련되어 있는 가장 최근 콘텐츠가 물리 블록 B2에 저장된다. 일반적으로, 물리 블록 B와 물리 블록 B1의 콘텐츠를 통합하는 것은 물리 블록 B와 물리 블록 B1 안에 있는 콘텐츠 그룹을 통합하는 것을 포함한다.

<109> 콘텐츠들이 물리 블록 B2에 통합된 후, 물리 블록 B1은 단계(1116)에서 삭제된다. 일단 물리 블록 B1이 삭제되면, 물리 블록 B1은 논리 블록 A로부터 단계(1118)에서 분리된다. 따라서, 프로세스 흐름은 단계(1122)로 진행하는데, 여기서 물리 블록 B가 삭제된다.

<110> 상기에서 설명된 RAM 캐시는 논리 그룹과 관련된 그룹 업데이트를 캐시하는데 사용될 수 있다. RAM 캐시는 상기 업데이트 프로세스가 완료되기까지 업데이트되고, 상기 시점에 논리 블록과 관련된 가장 최근의 데이터를 저장하기 위해 새로운 물리 블록이 얻어지고 이전 물리 블록이 삭제되므로, RAM 캐시의 사용은 일반적으로 논리 블록과 관련된 업데이트 프로세스의 과정 동안에 요구되는 삭제 프로세스의 수를 감소시킬 수 있다.

<111> 도12a를 참고하여, 그룹으로 효율적으로 분리되는 RAM 캐시의 사용은 본 발명의 일 실시예에 상응하여 설명된다. 논리 블록 A(100)는 논리 그룹(1204)으로 분리되며, 상기 논리 그룹 각각은 다수의 논리 페이지(1208)를 포함하고 있다. 물리 블록 B(1212)는 논리 블록 A(100)와 관련되어 있으며, 물리 블록 B(1212)는 논리 블록 A(1200)와 관련되어 있는 데이터 또는 콘텐츠를 포함한다. 도시되어 있는 것과 같이, 물리 블록 B(1212)의 물리 페이지를 포함하고 있는 제1 물리 그룹(1216a)은 논리 그룹(1204a)과 관련되어 있는 데이터(1220a)를 포함한다. 유사하게, 제2 물리 그룹(1216b)은 논리 그룹(1204b)과 관련되어 있는 데이터(1220b)를 포함하고 제3 물리 그룹(1216c)은 논리 그룹(1204c)과 관련되어 있는 데이터(1220c)을 포함하고 제4 물리 그룹(1216d)은 논리 그룹(1204d)과 관련되어 있는 데이터(1220d)을 포함한다.

업데이트(1224)가 예를 들어 호스트로부터 수신될 때, 이것은 논리 그룹(1204b)과 관련되어 있는 데이터가 업데이트된다는 것을 지시하며, 업데이트(1224)은 RAM 캐시(1228) 안에 있는 제1 가용 그룹(1232a)에 저장된다. 대안적으로, 일 실시예에서, 업데이트(1224)는 제2 논리 그룹(1204B)과 관련되어 있기 때문에, 업데이트(1124)은 제2 RAM 그룹(1232b)에 저장된다.

<112> 만약 추가의 업데이트가 논리 그룹(1204B)과 관련되어 있는 데이터가 다시 한번 업데이트된다는 지시를 수신하면, 상기와 같은 업데이트는 RAM 캐시(1228) 안에 있는 그룹(1232a)의 콘텐츠와 통합된다. 도12b에 도시되어

있는 것과 같이, 논리 그룹(1204b)에 상응하는 새로운 업데이트(136)이 수신될 때, 새로운 업데이트(1236)의 데이터는 논리 그룹(1204b)과 관련되어 있는 통합, 업데이트된 데이터(1224')을 형성하기 위해 RAM 그룹(1232a)에 이전에 저장되어 있는 콘텐츠와 통합될 수 있다. 일정한 경우에, RAM 은 재기록되기 때문에, 새롭게 업데이트된 데이터(1236) 중 일부는 이전에 업데이트된 데이터, 즉 도12a의 새로운 데이터(124)를 통합 프로세스 동안에 재기록할 수 있다.

<113> 논리 그룹(1204d)에 대한 업데이트(1240)이 이루어질 때, 업데이트(1240)은 다음 가용 RAM 페이지(1232b)에 저장될 수 있다. 업데이트(1236)이전에 또는 후에 업데이트(1240)이 수신되었는지에 따라 업데이트(1236)은 여전히 그룹(1232a)에 통합되어 있으며, 업데이트(1240)은 그룹(1232b)에 저장될 수 있다는 것을 이해할 수 있다.

<114> 일단 논리 블록 A(1200)에서 이루어진 모든 업데이트가 완료되면, 즉 또 다른 논리 블록이 업데이트될 때, RAM 캐시(1228)의 콘텐츠와 물리 블록 B(1212)의 콘텐츠는 새로운 물리 블록에 통합될 수 있다. 도12c는 본 발명의 실시예에 상응하여, 도12b의 논리 블록A(1200), 상기 RAM 캐시의 콘텐츠 이후의 새로운 물리 블록(1228) 및 상기 논리 블록과 관련되어 있는 최초 물리 블록(1212)의 계통도인데, 이들은 새로운 물리 블록으로 통합된다. 논리 블록 A(1200)에서 이루어진 업데이트가 완료되었다고 결정되면 물리 블록 B1(1236)은 전형적으로 얻어진다. 일단 물리 블록 B1(1236)이 얻어지면, RAM 캐시(1228)의 그룹(1232) 콘텐츠는 도10과 관련하여 설명한 것과 같이 물리 블록 B(1212)의 그룹(1220)의 콘텐츠와 통합되어 물리 블록 B1(1236)에 저장된다.

<115> 논리 그룹(1204b)과 연관되는 데이터는 RAM 캐시(1228)에서 업데이트되기 때문에, 업데이트된 데이터(1224')은 논리 그룹(1204b)과 관련되어 있는 최초 데이터를 나타내는 데이터(1220b)과 통합된다. 데이터(1220B)은 논리 그룹(1204)과 관련되어 있는 현재의 콘텐츠를 포함하며, 데이터(1220B)은 모두 업데이트된 데이터(1224')의 관점에서는 이전 것이다. 논리 그룹(1204B)과 관련되어 있는 현재 데이터 또는 가장 최근 데이터가 물리 블록 b1(1236)에서 나타나도록 하기 위해, 통합 프로세스는 업데이트된 데이터(1224')와 데이터(1220b)로부터 통합된 데이터(1244)를 생성하는데 사용된다. RAM 캐시(1228)의 RAM 그룹(1232)에 포함되어 있는 데이터는 전형적으로 동일한 논리 그룹(1204)과 관련되어 있는 동일한 페이지 오프셋이 상응하는 물리 그룹(1216) 안에 데이터를 포함하고 있을 때, 통합된 데이터에 포함되어 있다. 즉, 논리 그룹 안에 있는 게이트 오프셋에 상응하는 데이터가 물리 그룹(1216)과 RAM 그룹(1232)에 존재할 때, RAM 그룹(1232)에 있는 데이터는 보다 최근이기 때문에 RAM 그룹(1232)에 있는 데이터는 통합된 데이터에 포함되어 있다.

<116> 설명되는 실시예에서 논리 그룹(1204a)과 관련되어 있는 최초 데이터인 데이터(1220a)이 물리 그룹(1240a)에 저장된 후에, 통합된 데이터(1244)는 제2 물리 그룹(1204b)에 저장된다. 논리 그룹(1204c)과 관련되어 있는 데이터(1220c)은 업데이트되지 않기 때문에, 데이터(1220c)은 물리 그룹(1240c)에 저장된다. 논리 그룹(1204d)이 RAM 캐시(1228)의 RAM 그룹(1232b)에 저장되어 있는 새로운 또는 업데이트된 데이터(1240)뿐만 아니라 물리 블록B(1212)의 물리 그룹(1216d)에 저장되어 있는 관련된 최초 데이터(1220d)를 가지고 있다. 따라서, 통합 프로세스는 논리 그룹(1204d)과 관련된 가장 최근 또는 현재의 콘텐츠를 나타내는 통합된 데이터(1248)을 생성하기 위해 데이터(1220d)과 새로운 도는 업데이트된 데이터(1240)을 사용하여 수행된다. 도시되어 있는 것과 같이, 통합된 데이터(1248)는 물리 그룹(1240d)에 저장되어 있다. 따라서, 물리 블록 A(1200)와 관련되어 있는 현재의 콘텐츠를 물리 블록 B1(1236)가 포함하고 있기 때문에, 물리 블록 B(1212)는 이전 것이 되며, 논리 블록 A(1200)으로부터 분리될 뿐만 아니라 삭제된다.

<117> 도12d는 본 발명의 일 실시예에 상응하여, 논리 블록, 물리 블록 및 RAM 캐시의 계통도인인데, 새로운 데이터를 수용하기에 충분한 여유가 물리 블록에 있지 않기 때문에 상기 RAM 캐시의 콘텐츠가 RAM 캐시에 존재한다. 물리 블록(1212)의 그룹(1216)은 그룹(1216a)이 논리 그룹(1204b)과 관련되어 있는 최초 데이터(1220a)을 포함하고, 그룹(1216b)이 논리 그룹(1204b)과 관련되어 있는 업데이트된 데이터(120a')을 포함하고, 그룹(1216c)은 논리 그룹(1204c)과 관련되어 있는 데이터(1220c)를 포함하며, 그룹(1216d)은 논리 그룹(1204d)과 관련되어 있는 데이터(1220a'')을 포함한다. 따라서, 물리 블록(1212)은 추가의 데이터가 기록될 수 있는 가용 그룹(1216)이 존재하지 않고 완전히 채워져 있다.

<118> 논리 그룹(1204d)에 상응하는 새로운 데이터(1240)가 제공되면, 새로운 데이터(1240)는 RAM 캐시(1228)의 RAM 그룹(1232)에 적절히 저장된다. 일 실시예에서, 상기 적절한 RAM 그룹(1232)은 RAM 캐시(1228) 안에 있는 제1 가용 RAM 그룹(1232a)일 수 있다. 논리 블록 A(100)로 이루어지는 추가의 업데이트가 더 이상 없다고 결정이 될 때, 새로운 물리 블록 B1(1252)는 논리 블록 A(1200)와 관련되어 얻어진다.

<119> 물리 블록 B1(1252) 안에서, 제1 물리 그룹(1260)은 논리 그룹(1204b)과 관련되어 있는 통합된 데이터(1260)을 포함한다. 통합된 데이터(1260)는 데이터(1220a, 1220a', 120a'')을 통합함으로써 생성되는데, 논리 그룹

(1204b)과 관련되어 있는 가장 최근의 데이터는 통합된 데이터(1260)에 포함되어 있다. 논리 그룹(1204b)과 관련된 가장 최근의 데이터가 통합된 데이터(1260)에 포함되는 것을 보장하기 위해, 그룹(1216)은 그룹(116d)에서 시작하여 그룹(1216a)으로 끝나도록 프로세스되며, 따라서 소정 논리 그룹에 관련되어 있는 특정 페이지 오프셋에 포함되어 있는 가장 최근의 데이터는 쉽게 식별된다.

<120> 일단 통합된 데이터(1260)이 생성되고 그룹(1256a)에 저장되면, 논리 그룹(1204c)과 관련되어 있는 데이터(1220c)은 그룹(1256b)에 저장되며, RAM 캐시(1228)에 있는 데이터(1240)는 그룹(1256)에 저장된다. 결국, 물리 블록 B1(1252)은 논리 블록 A(1200)과 관련되어 있는 가장 최근 데이터를 포함한다.

<121> 도13을 참고하여, 본 발명의 일 실시예에 상응하여, 논리 블록의 논리 그룹에 관련되어 있는 업데이트를 저장하는데 RAM 캐시를 사용하는 한 방법에 관련된 단계들이 설명되어 있다. 프로세스(1300)는 단계(1304)에서 시작되는데, 사기 단계에서 호스트는 물리 블록 B와 관련되어 있는 논리 블록 A를 프로세스하기 시작한다. 논리 블록 A를 프로세스하는 것은 일반적으로 논리 블록 A에 있는 논리 그룹들로의 업데이트를 처리하는 단계를 포함한다.

<122> 단계(1308)에서, RAM 캐시는 클리어된다. 일단 RAM 캐시가 클리어되면, 논리 블록 A와 관련되어 있는 업데이트된 콘텐츠는 단계(1312)에서 RAM 캐시에 저장된다. 특히, 업데이트된 콘텐츠는 상기 업데이트된 콘텐츠와 관련되어 있는 논리 그룹에 근거하여 적절한 RAM 그룹에 저장될 수 있다. 따라서, 단계(1316)에서, 저장되는 논리 블록 A와 관련된 추가의 업데이트 콘텐츠가 존재하는지에 대해 결정한다. 만약 저장할 추가의 업데이트된 콘텐츠가 있다고 결정되면, 프로세스 흐름은 업데이트된 콘텐츠가 저장되는 단계(1312)로 변환된다.

<123> 대안적으로, 논리 블록 A를 위해 저장된 추가의 업데이트된 콘텐츠가 존재하지 않다는 결정이 단계(131)에서 이루어지면, 논리 블록 A와 관련되어 단계(1320)에서 새로운 물리 블록 B1이 할당된다. 달리 말하면, 논리 블록 A와 관련된 가장 최근 데이터가 기록되는 물리 블록 B1이 얻어진다. 일단 새로운 물리 블록 B1이 할당되면, RAM 캐시의 콘텐츠와 물리 블록 B의 콘텐츠는 단계(1324)에서 적절하게 물리 블록 B1에 통합된다. 상기 콘텐츠를 통합하는 단계는 일반적으로 각각의 논리 그룹과 관련된 콘텐츠를 통합하는 단계를 포함하며, 따라서 통합된 콘텐츠는 각 논리 그룹과 관련된 가장 최근 또는 현재의 콘텐츠이다.

<124> 일단 RAM 캐시의 콘텐츠와 물리 블록 B의 콘텐츠가 통합되면, 물리 블록 B1은 논리 블록 A와 관련된 가장 최근의 콘텐츠를 포함한다. 따라서, 단계(1328)에서, 물리 블록 B는 삭제된다. 물리 블록 B가 삭제된 후, 물리 블록 B는 논리 블록 A로부터 분리되며, 업데이트를 저장하기 위해 RAM 캐시를 사용하는 프로세스는 완료된다.

<125> 일정한 경우에, 예를 들어, 시스템 내에 충분한 가용 RAM이 존재하지 않으면 상기 RAM 캐시의 사용은 불가능하다. RAM 캐시가 가용되지 않을 때, 캐시 블록 또는 블록 캐시는 논리 블록과 관련된 그룹 업데이트를 저장하는데 대신 사용될 수 있다. 본 발명의 일 실시예에 상응하여, 그룹으로 분리되는 논리 블록에 관련되어 있는 업데이트를 저장하는데 적당히 사용되며, 그룹으로 효율적으로 분리되는 블록 캐시의 계통도이다. 논리 블록 A(1400)는 논리 그룹(1404)으로 분리되며, 각각의 논리 그룹은 다수의 논리 페이지(1408)를 포함한다. 물리 블록 B(1412)는 논리 블록 A(4200)와 관련되며, 따라서 물리 블록 B(1412)는 논리 블록 A(1400)와 관련된 데이터를 포함하고 있다. 물리 블록 B(1412)의 물리 페이지를 포함하고 있는 제1 물리 그룹(1416a)은 논리 그룹(1404a)과 관련된 데이터(1420a)를 포함하고 있다. 유사하게, 제4 물리 그룹(1416d)은 논리 그룹(1404b)과 관련된 업데이트된 데이터(1420d)를 포함하고 있다. 제2 물리 그룹(1416b)은 논리 그룹(1404a)과 관련된 데이터(1420b)를 포함하고 있으며 제3 물리 그룹(1416c)은 논리 그룹(1204c)과 관련된 데이터(1420c)을 포함하고 있다. 논리 블록 B(1412)은 논리 블록(1404d)과 관련된 어떠한 데이터도 포함하지 않는다.

<126> 업데이트(1424)이 논리 그룹(1404d)과 관련된 데이터를 포함하고 있다는 것을 수신할 때, 업데이트 (1424)은 물리 블록 캐시(1428) 안에 있는 제1 가용 그룹(1432a)에 저장된다. 만약 추가의 업데이트가 논리 그룹(1404d)과 관련된 데이터가 다시 한번 업데이트된다는 지시를 수신하면, 상기와 같은 업데이트는 블록 캐시(1428)안에 있는 다음 가용 그룹(1432b)에 저장될 수 있다. 도14b에 도시되어 있는 것과 같이, 논리 그룹(1404d)에 상응하는 새로운 업데이트(1426)이 수신될 때, 새로운 업데이트(1426)에 있는 데이터는 그룹(1432b)에 저장된다. 일 실시예에서, 데이터(1424, 1426) 모두에 포함되어 있는 논리 그룹(1404d)과 관련된 일정한 현재 데이터가 존재할 수 있다는 것을 이해하여야 한다. 그러나, 정보 페이지가 데이터(1426)에 존재하지 않을 때, 만약 데이터(1424)의 상응하는 페이지가 정보를 포함하고 있다면, 데이터(1426)의 정보는 전형적으로 논리 그룹(1404d)과 관련되어 있는 가장 최근 데이터가 요구될 때 사용될 것이다.

<127> 도14c는 논리 블록 즉, 도14b의 논리 블록 A(1400), 블록 캐시 및 논리 블록과 관련되어 있는 최초 물리 블록,

즉 도14b의 블록 캐시(1428)와 물리 블록 B(1412)의 켄텐츠 후의 새로운 물리 블록의 계통도인데, 상기 블록 캐시와 물리 블록은 상기 새로운 불리 블록에 통합된다. 새로운 블록 B1(1440)은 논리 블록 A(1400)로 이루어지는 추가의 업데이트가 없을 때 논리 블록 A(1400)와 관련되도록 얻어진다. 논리 블록 A(1400)와 관련된 업데이트가 완료될 때, 블록 캐시(1428)에 있는 가용 그룹(1432c, 1432d)은 효율적으로 비워지며, 새로운 또는 다른 논리 블록의 논리 그룹과 관련된 데이터를 저장할 수 있으며, 또는 논리 블록 A(1400)에 나중에 저장될 수 있다.

<128> 일단 새로운 물리 블록 B1(1440)이 얻어지면, 논리 그룹(1404b)과 관련된 데이터(1420a, 1420d)은 논리 블록 B1(1440)의 새로운 그룹(1444a)에 저장될 수 있는 통합된 데이터(1446)를 형성하도록 도10에 도시되어 있는 것과 같이 통합된다. 데이터(1420b)는 그룹(1416b)로부터 그룹(1444b)으로 복사되며, 데이터(1420c)은 그룹(1416c)로부터 그룹(1444c)으로 복사될 수 있다.

<129> 블록 캐시(1428)은 논리 그룹(1404d)과 관련된 데이터(1424, 1426)를 포함하기 때문에, 데이터(1424, 1426)는 새로운 물리 블록 B1(1440)의 물리 그룹(1444d)에 저장되는 통합된 데이터(1448)를 형성하도록 통합된다. 통합된 데이터(1448)는 일반적으로 논리 그룹(1404d)과 관련된 가장 최근 또는 최근의 콘텐츠를 포함한다. 일단 새로운 물리 그룹 B1(1440)이 채워지면, 새로운 물리 블록 B1(1440)은 효율적으로 논리 블록 A(1400)와 관련된 가장 최근 또는 최근의 콘텐츠를 포함한다. 따라서, 물리 블록 B(1412)는 이전 것이 되며, 삭제되고 논리 블록 A(1400)으로부터 분리된다.

<130> 이전에 언급한 것과 같이, 블록 캐시(1428)는 다른 논리 블록 또는 논리 블록 A(1400)와 관련된 새로운 데이터 또는 업데이트를 연속하는 업데이트 프로세스 동안에 저장하는데 사용되는 가용 그룹(1432)을 포함한다. 도14d를 참고하여, 논리 블록 A(1400)와 관련된 새로운 업데이트를 저장하는 블록 캐시(1428) 안에 있는 가용 그룹(1432c, 1432d)의 사용을 본 발명의 일 실시예에 상응하여 설명하고 있다. 블록 캐시(1428) 안에 있는 그룹(1432a, 1432b)의 콘텐츠는 어는 일정한 논리 블록과 관련된 이전 콘텐츠일 수 있으며, 논리 블록 A(1400)와 관련된 이전 콘텐츠에 한정되지 않는다는 것을 이해하여야 한다. 전형적으로, 그룹(1432a, 1432b)의 이전 콘텐츠는 다른 논리 블록에 관련되어 있다.

<131> 논리 그룹(1404d)와 관련된 업데이트가 수신될 때, 업데이트의 콘텐츠는 논리 그룹(1404d)에 상응하는 이전 데이터(1422a)와 통합될 수 있으며, 물리 블록(1412)의 물리 그룹(1416a')안에 저장된다. 논리 그룹(1404d)과 관련된 통합된 데이터(1450)는 블록 캐시(1428) 안에 있는 1 가용 그룹(1432c)에 저장된다. 유사하게, 논리 그룹(1404b)과 관련된 업데이트가 수신될 때, 업데이트 콘텐츠는 논리 그룹(1404b)에 상응하는 물리 그룹(1416c')에 저장되어 있는 이전 데이터(1422c)와 통합된다.

<132> 논리 블록 A(1400)와 관련된 추가 정보가 캐시되는 것이 요구될 때, 블록 캐시(1428)은 완전히 채워져 있기 때문에, 새로운 블록 캐시가 얻어진다. 도14e를 참고하여, 논리 블록 A(1400)와 관련된 업데이트를 저장하는 2개의 블록 캐시의 사용이 본 발명의 일 실시예에 상응하여 설명된다. 새로운 또는 최근 블록 캐시(1464)는 블록 캐시(1428)가 완전히 채워질 때 그리고 논리 블록 A(1400)의 업데이트가 수신될 때, 얻어질 수 있다. 예를 들어, 논리 그룹(1404c)와 관련된 업데이트가 수신될 때, 데이터(1422b)는 또한 논리 그룹(1404c)과 관련되어 있기 때문에, 업데이트과 관련된 데이터는 그룹(1416b)에 포함되어 있는 데이터(1422b)에 통합된다. 일단 통합된 데이터가 생성되면, 통합된 데이터(1460)는 최근 블록 캐시(1464)의 제 1 가용 그룹(1468)에 저장된다.

<133> 일반적으로, 최근 캐시 블록(1464)은 최근 캐시 블록(1464)이 추가의 가용 그룹을 가지고 있지 않을 때가지 업데이트 또는 통합된 업데이트를 적절히 저장하는데 사용될 수 있다. 최근 캐시 블록(1464)이 완전히 채워질 때, 그리고 논리 블록A(1400)과 관련된 추가의 업데이트가 요구될 때, 새로운 최근 캐시 블록이 얻어진다.

<134> 만약 통합된 데이터(1460)가 제1 가용 그룹(1468)에 저장된 후, 논리 블록A(1400)와 관련된 추가의 업데이트가 존재하지 않을 경우, 물리 블록 b(1412)와 블록 캐시(1428)의 콘텐츠를 최근의 블록 캐시(1464)에 통합하여야 하는지와 물리 블록 B(1412), 물리 캐시(1428) 및 최근 블록 캐시(1464)의 콘텐츠가 새로운 물리 블록에 통합되어야 하는지에 대해 결정한다. 도시된 실시예에서, 물리 블록 B(1412)와 블록 캐시(1428)의 콘텐츠를 최근 블록 캐시(1464)에 통합할 수 있다. 도14f는 본 발명의 일 실시예에 상응하여, 블록 캐시(1428)와 물리 블록 B(1412)의 콘텐츠가 최근 블록 캐시(1464)에 통합된 후, 최근의 블록 캐시(1464)의 계통도이다. 최근 블록 캐시는 이미 논리 그룹(1404c)과 관련된 데이터(1460)에 포함되어 있다. 블록 캐시(1428)는 논리 그룹(1404d)와 관련된 통합 데이터(1450)와 논리 그룹(1404b)와 관련된 통합 데이터(1452)를 포함하는데, 즉 블록 캐시(1428)는 이미 논리 그룹(1404b, 1404c)과 관련된 가장 최근 데이터를 포함하기 때문에, 통합된 데이터(1450)는 통합된 데이터(1452)가 그룹(1468c)에 복사될 수 있는 반면 그룹(1468b)에 복사될 수 있다. 마지막으로, 논리 그룹

(1404a)과 관련되어 있으며, 물리 블록 B(1412)의 그룹(1416d')에 포함되어 있는 최초 데이터(1422d)는 그룹(1468d)에 복사될 수 있으며, 따라서 효율적으로 논리 블록 A(1400)된 가장 최근 콘텐츠를 최근 블록 캐시(1464)에 통합한다. 그 다음, 현재의 블록 캐시(1464)는 논리 블록 'A'(1400)와 연관되는 새로운 물리 블록으로서 논리 블록 'A'(1400)와 연관될 수 있다는 점을 인식해야 한다.

<135> 도15a, 15b를 참고하여, 본 발명의 일 실시예에 상응하여, 그룹을 사용하는 블록 캐시의 업데이트된 콘텐츠를 저장하는 한 방법과 관련된 단계가 설명되어 있다. 프로세스(1500)는 단계(1504)에서 시작되는데, 호스트는 논리 블록 A를 프로세스하기 시작하는데, 예를 들어, 물리 블록 B와 관련되어 있는 논리 블록 A의 업데이트를 프로세스한다. 단계(1508)에서, 논리 그룹A와 관련된 최근 블록이 존재하는지 그리고 최근 캐시 블록에 가용 여유가 있는지에 대하여 결정한다.

<136> 만약, 단계(1508)에서 최근 캐시 블록이 존재하지 않거나 또는 최근 캐시 블록이 여유 공간을 가지고 있지 않다고 결정되면, 새로운 퇴근 캐시 블록이 단계(1512)에서 얻어진다. 일단 상기 새로운 최근 캐시 블록이 얻어지면, 프로세스 흐름은 단계(1516)로 이동하는데, 여기서 물리 블록 B안에 있는 물리 그룹에 저장되어 있는 소정의 논리 그룹에 관련된 데이터를 통합함으로써 생성된 콘텐츠와 상기 소정 논리 그룹과 관련된 새로운 데이터가 최근 캐시 블록에 저장된다. 만약 단계(1508)에서 상기 최근 캐시 블록이 가용 여유를 가지고 있다고 결정되면, 프로세스 흐름은 직접 단계(1508)에서 단계(1516)로 진행된다.

<137> 일단 통합된 콘텐츠가 상기 최근 캐시 블록에 저장되면, 단계(1520)에서 논리 블록 A에 저장되는 추가의 업데이트된 콘텐츠가 있는지 결정된다. 만약 저장되어야 하는 추가의 업데이트 콘텐츠가 존재한다고 결정되면, 프로세스 흐름은 단계(1508)로 반환되는데, 여기서 최근 캐시 블록에 가용 여유가 있는지 결정된다. 대안적으로, 논리 블록 A에 대해 저장되는 추가의 업데이트 콘텐츠가 존재한다고 결정된다면, 물리 블록 B 또는 어느 일정한 더 이전의 캐시 블록에 포함되어 있는 정보를 최근 캐시 블록에 통합하는 최근 캐시 블록에 적절한 여유가 있는지에 대하여 단계(1524)에서 결정된다.

<138> 만약 최근 캐시 블록에 통합을 수행하는 적절한 여유가 있다고 결정되면, 논리 블록 A와 관련된 새로운 물리 블록을 형성하기에 최근의 캐시 블록이 적절하다고 지시하며, 따라서 논리 블록 A에 관련된 모든 최근 데이터를 포함하기에 적절하다. 따라서, 단계(1536)에서, 물리 블록 B 안에 있는 그룹 콘텐츠와 어는 일정한 이전 캐시 블록의 그룹 콘텐츠는 적절히 최근 캐시 블록에 통합되며, 따라서 상기 최근 캐시 블록은 논리 블록 A안에 있는 그룹과 관련된 가장 최근 데이터를 포함한다.

<139> 최근 캐시 블록으로 통합하는 프로세스가 완료된 후, 물리 블록 B 또는 다른 이전 캐시 블록은 단계(1540)에서 삭제된다. 일단, 물리 블록 B가 삭제되면, 물리 블록 B는 논리 블록 A와 단계(1544)에서 분리되며, 상기 통합된 최근 캐시 블록은 단계(1548)에서 논리 블록 A와 관련되며, 따라서 논리 블록 A와 관련된 물리 블록을 형성하며, 논리 블록 A와 관련된 가장 최근 콘텐츠를 포함한다. 일단 상기 통합된 캐시 블록이 논리 블록 A와 관련되면, 블록 캐시의 업데이트 콘텐츠를 저장하는 프로세스가 완료된다.

<140> 단계(1524)로 반환되면서, 최근 캐시 블록으로의 통합이 수행되지 않는다고 결정되면, 최근 캐시 블록에 나타나지 않는 논리 그룹과 관련된 데이터를 수용하기 위해 상기 최근 캐시 블록 안의 물리 그룹의 수가 충분하지 않다고 지시한다. 따라서, 단계(1532)에서, 새로운 물리 블록 B1은 논리 블록 A를 위해 할당된다. 단계(1532)에서, 물리 블록 B 안에 있는 그룹 콘텐츠는 모든 캐시 블록의 그룹 콘텐츠와 함께 물리 블록 B1로 통합된다. 단계(1532)로부터, 프로세스 흐름은 단계(1540)로 진행되는데, 물리 블록 B와 캐시 블록은 삭제된다. 통합된 캐시 블록이 논리 블록 A와 관련되어 있는 단계(1548)는 물리 블록 B1이 논리 블록 A와 관련될 때 수행되지 않는다.

<141> 비록 단지 본 발명의 수 개의 실시예만이 설명되었지만, 본 발명의 범위 안에서 다른 실시예를 가지고 있다는 것을 이해하여야 한다. 예로써, 업데이트된 데이터를 일시적으로 저장하거나 또는 캐시하는 RAM 캐시 및 물리 블록의 사용이 설명되었지만, 어느 일정한 메모리가 업데이트된 데이터를 캐시하는데 사용된다. 달리 말하면, 논리 블록에 맵되는 물리 블록에 업데이트를 기록하기 전에 논리 블록과 관련된 업데이트를 저장하는 캐시는 RAM 캐시 또는 물리 캐시에 한정되지 않는다.

<142> 플래시 메모리의 사이즈와 플래시 메모리에 있는 블록의 사이즈는 넓게 변화할 수 있다. 블록 사이즈는 일반적으로 32 페이지를 포함하도록 설명되었지만, 블록은 다른 수의 페이지 또는 블록을 구성하는 더 많은 수의 엘리먼트를 포함할 수 있다. 예를 들어, 블록은 64페이지 또는 엘리먼트를 포함할 수 있다. 결국, 시스템 안에 있는 블록의 수는 변화될 수 있다. 512Mb NAND 플래시 메모리 안에, 블록이 각각 512 바이트를 포함하고 있는 32

개의 페이지를 포함하고 있다면, 총 4096 물리 블록이 플래시 메모리에 있을 수 있다. 대안적으로, 동일한 51Mb NAND 플래시 메모리 안에, 각각의 물리 블록이 512바이트를 각각 포함하고 있는 64개의 페이지를 포함하고 있다면, 총 2048 물리 블록이 플래시 메모리에 있을 수 있다. 일반적으로, 페이지 사이즈는 변화할 수 있다.

<143> 본 발명의 여러 방법과 관련된 단계들은 넓게 변화될 수 있다. 일반적으로, 단계들이 추가되거나, 제거되거나, 순서가 바뀌거나 그리고 변경될 수 있다. 예로써, 본 발명의 범위 안에서 물리 블록 캐시를 사용하는 방법은 물리 블록 캐시가 완전히 채워질 때를 결정하는 단계, 물리 블록 캐시로의 업데이트를 계속해서 캐시하기 전에 물리 블록 캐시와 물리 블록 A의 콘텐츠를 물리 블록 B에 결합하는 단계를 포함할 수 있다. 게다가, 업데이트 또는 업데이트가 캐시되기 전에 예를 들어, 새로운 물리 블록이 RAM 캐시의 콘텐츠를 물리 블록에 복사하여야 할 때 또는 이전 물리 블록과 물리 블록 캐시의 콘텐츠를 통합할 때 반드시 새로운 물리 블록이 얻어져야 하는 것은 아니다. 따라서, 본 실시예는 단지 설명을 위한 것이며, 본 발명은 첨부된 청구항들의 범위에서 변경될 수 있다.

발명의 효과

<144> 본 발명은 논리 페이지 업데이트에 응답하여 발생하는 기록 및 삭제 프로세스의 회수를 줄이는 방법 및 장치를 제공한다. 특히, 논리 블록 도메인의 업데이트에 상응하는 물리 블록 도메인의 업데이트를 수행하는 것과 관련된 기록 및 삭제 작동 용량을 물리 블록 도메인의 업데이트를 수행하는 능력 감소 없이 줄이는 시스템을 제공한다.

도면의 간단한 설명

<1> 본 발명은 이하의 도면과 결합한 상세한 설명을 통해 잘 이해할 수 있을 것이다.

<2> 도1은 상기 업데이트된 콘텐츠를 수용하기 위해 얻어질 수 있는 물리 블록과 업데이트된 콘텐츠를 가지고 있는 논리 블록을 도시하고 있는 계통도이다.

<3> 도2는 논리 블록과 관련된 페이지의 업데이트된 콘텐츠를 물리 블록에 기록하는 하나의 방법과 관련된 단계를 설명하는 흐름도이다.

<4> 도3a는 비휘발성 메모리를 포함하고 있는 일반적인 호스트 시스템의 계통도이다.

<5> 도3b는 도3a의 메모리 소자(120)의 메모리 소자의 계통도이다.

<6> 도3c는 내장형 비휘발성 메모리를 포함하고 있는 호스트 시스템의 계통도이다.

<7> 도4a는 본 발명의 실시예에 상응하는 논리 블록에 관련된 업데이트를 캐시하도록 구성된 RAM 캐시, 논리 블록 및 물리 블록의 계통도이다.

<8> 도4b는 본 발명의 실시예에 상응하여 도4a의 논리 블록(400), 물리 블록(410) 및 RAM 캐시(420)의 계통도인데, 여기서 상기 RAM 캐시는 논리 블록과 관련되어 있는 다수의 업데이트를 캐시한다.

<9> 도4c는 본 발명의 실시예에 상응하게 도4b의 물리 블록(410) 및 RAM 캐시(420)의 계통도인데, 새로운 물리 블록으로 기록되는 콘텐츠를 가지고 있다.

<10> 도5는 본 발명의 일 실시예에 상응하여 RAM 캐시를 사용하여 물리 블록 도메인의 페이지 콘텐츠를 업데이트하는 한 방법에 관련된 단계를 설명하는 흐름도이다.

<11> 도6a는 본 발명의 일 실시예에 상응하여 논리 블록에 관련된 업데이트를 캐시하도록 구성된 물리 블록 캐시, 논리 블록 및 물리 블록의 계통도이다.

<12> 도6b는 도6a의 물리 블록(610) 및 물리 블록 캐시(612)의 계통도인데, 본 발명의 일 실시예에 상응하여 새로운 물리 블록으로 통합되는 콘텐츠를 가지고 있다.

<13> 도7은 본 발명의 일 실시예에 상응하여 물리 블록을 사용하여 물리 블록 도메인의 페이지 콘텐츠를 업데이트하는 한 방법에 관련된 단계를 설명하는 흐름도이다.

<14> 도8은 본 발명의 일 실시예에 상응하여 시스템 구조의 계통도이다.

<15> 도9a는 본 발명의 일 실시예에 상응하여 그룹으로 분리되는 논리 블록, 물리 블록 및 새로운 물리 블록의 계통도이다.

<16> 도9b는 본 발명의 일 실시예에 상응하게 도9a의 논리 블록(900), 물리 블록(912) 및 새로운 물리 블록(928)의 계통도인데, 여기서 상기 새로운 물리 블록은 상기 논리 블록과 관련된 가장 최근의 콘텐츠를 포함하고 있다.

<17> 도9c는 본 발명의 일 실시예에 상응하게 새로운 물리 블록뿐만 아니라 도9b의 논리 블록(900) 및 물리 블록(928)의 계통도인데, 상기 새로운 물리 블록은 논리 블록과 관련되어 있는 가장 최근의 콘텐츠를 포함하고 있다.

<18> 도9d는 본 발명의 일 실시예에 상응하게 그룹으로 분리되는 논리 블록, 상기 논리 블록과 관련된 업데이트된 데이터를 포함하고 있는 물리 블록 및 상기 논리 블록에 관련된 가장 최근 데이터를 포함하고 있는 새로운 물리 블록의 계통도이다.

<19> 도10은 본 발명의 일 실시예에 상응하게, 최초 데이터 그룹, 새로운 또는 업데이트된 상기 그룹을 위한 데이터 및 최초 데이터 및 새로운 데이터 또는 업데이트된 데이터의 혼합을 도시하고 있는 계통도이다.

<20> 도11a과 11b는 본 발명의 일 실시예에 상응하는, 그룹으로 분리되는 논리 블록에 관련되어 있는 상기 콘텐츠를 업데이트하는 한 방법에 관련되어 있는 단계를 설명하는 흐름도이다.

<21> 도12a는 본 발명의 일 실시예에 상응하는, 그룹으로 분리되는 논리 블록과 관련되어 있는 업데이트를 저장하는데 적당하며 효율적으로 그룹으로 분리하는 RAM 캐시에 관하 계통도이다.

<22> 도12b는 본 발명의 일 실시예에 상응하는, 도12a의 RAM 캐시(1228)의 계통도인데, 여기서 동일한 논리 그룹으로의 반복된 업데이트는 상기 동일한 RAM 그룹으로 저장된다.

<23> 도12c는 본 발명의 실시예에 상응하여, 도12b의 논리 블록A(1200), 상기 RAM 캐시의 콘텐츠 이후의 새로운 물리 블록(1228) 및 상기 논리 블록과 관련되어 있는 최초 물리 블록(1212)의 계통도인데, 이들은 새로운 물리 블록으로 통합된다.

<24> 도12d는 본 발명의 일 실시예에 상응하여, 논리 블록, 물리 블록 및 RAM 캐시의 계통도인인데, 새로운 데이터를 수용하기에 충분한 여유가 물리 블록에 있지 않기 때문에 상기 RAM 캐시의 콘텐츠가 RAM 캐시에 존재한다.

<25> 도13은 본 발명의 일 실시예에 상응하여, 논리 블록의 논리 그룹에 관련되어 있는 업데이트를 저장하는데 RAM 캐시를 사용하는 한 방법을 설명하는 흐름도이다.

<26> 도14a는 본 발명의 일 실시예에 상응하여, 그룹으로 분리되는 논리 블록에 관련되어 있는 업데이트를 저장하는데 적당히 사용되며, 그룹으로 효율적으로 분리되는 블록 캐시의 계통도이다.

<27> 도14b는 본 발명의 일 실시예에 상응하여, 도14a의 블록 캐시(1428)의 계통도인데, 동일한 논리 그룹으로의 업데이트가 저장된다.

<28> 도14c는 논리 블록 즉, 도14b의 논리 블록 A(1400), 블록 캐시 및 논리 블록과 관련되어 있는 최초 물리 블록, 즉 도14b의 블록 캐시(1428)과 물리 블록 B(1412)의 켄텐츠 후의 새로운 물리 블록의 계통도인데, 상기 상기 블록 캐시와 물리 블록은 상기 새로운 불리 블록에 통합된다.

<29> 도14d는 본 발명의 일 실시예에 상응하여, 비가용 그룹을 포함하고 있는 도14c의 블록 캐시(1428)의 계통도인데, 상기 그룹에는 논리 블록 즉, 도14c의 논리 블록 A(1400)에 관련되어 있는 업데이트는 저장될 수 있다.

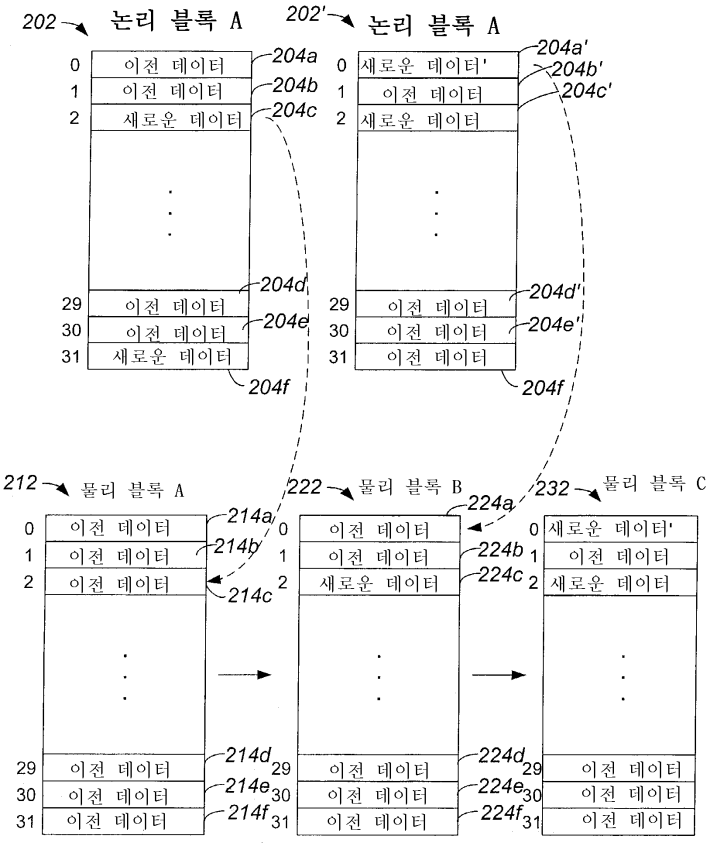
<30> 도14e는 본 발명의 일 실시예에 상응하여, 논리 블록 예를 들어 도 14d의 논리 블록 A(1400)과 관련되어 있는 업데이트를 저장하는데 사용될 수 있는 2개의 블록 캐시에 대한 계통도이다.

<31> 도14f는 본 발명의 일 실시예에 상응하여, 블록 캐시의 콘텐츠와 물리 블록의 콘텐츠, 예를 들어 도14e의 블록 캐시(1428)과 물리 블록 B(1412)의 콘텐츠가 현재의 블록 캐시에 통합된 후의 현재의 블록 캐시를 도시하고 있는 계통도이다.

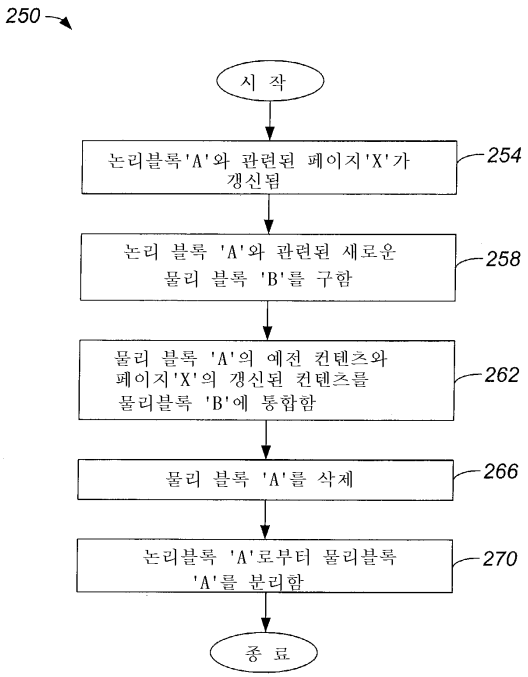
<32> 도15a와 도15b는 본 발명의 일 실시예에 상응하여, 그룹을 사용하는 블록 캐시의 업데이트된 콘텐츠를 저장하는 한 방법과 관련되어 있는 단계를 설명하고 있는 흐름도이다.

도면

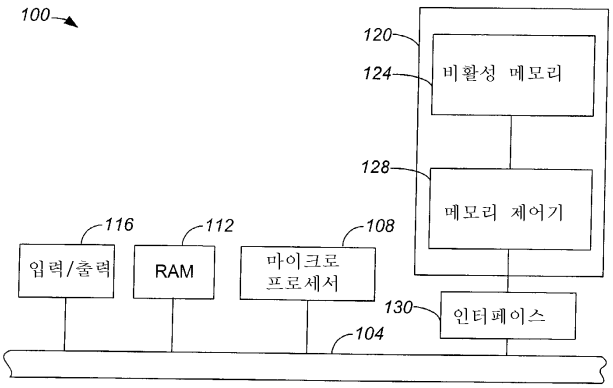
도면1


202
논리 블록 A
0 이전 데이터
1 이전 데이터
2 새로운 데이터
29 이전 데이터
30 이전 데이터
31 새로운 데이터
204a
204b
204c
204d
204e
204f
202'
논리 블록 A
0 새로운 데이터'
1 이전 데이터
2 새로운 데이터
29 이전 데이터
30 이전 데이터
31 이전 데이터
204a'
204b'
204c'
204d'
204e'
204f
212
물리 블록 A
0 이전 데이터
1 이전 데이터
2 이전 데이터
29 이전 데이터
30 이전 데이터
31 이전 데이터
214a
214b
214c
214d
214e
214f
222
물리 블록 B
0 이전 데이터
1 이전 데이터
2 새로운 데이터
29 이전 데이터
30 이전 데이터
31 이전 데이터
224a
224b
224c
224d
224e
224f
232
물리 블록 C
0 새로운 데이터'
1 이전 데이터
2 새로운 데이터
29 이전 데이터
30 이전 데이터
31 이전 데이터

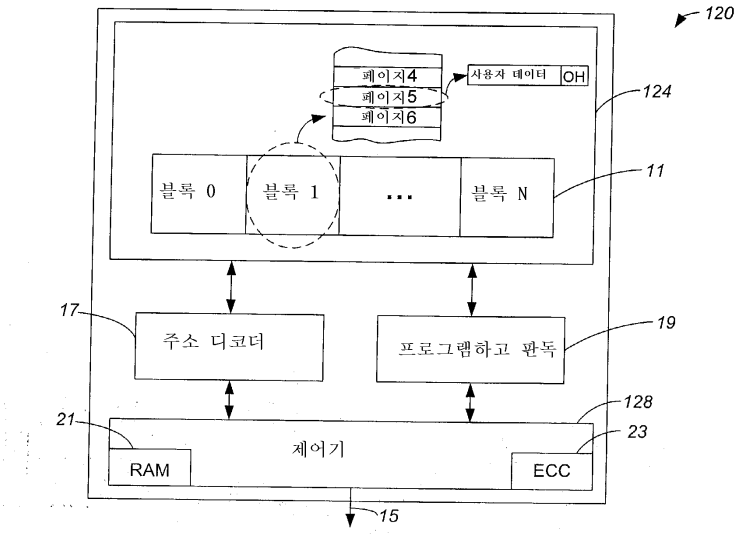
도면2


250
시 작
논리블록'A'와 관련된 페이지'X'가 갱신됨
254
논리 블록 'A'와 관련된 새로운 물리 블록 'B'를 구함
258
물리 블록 'A'의 예전 컨텐츠와 페이지'X'의 갱신된 컨텐츠를 물리블록 'B'에 통합함
262
물리 블록 'A'를 삭제
266
논리블록 'A'로부터 물리블록 'A'를 분리함
270
종 료

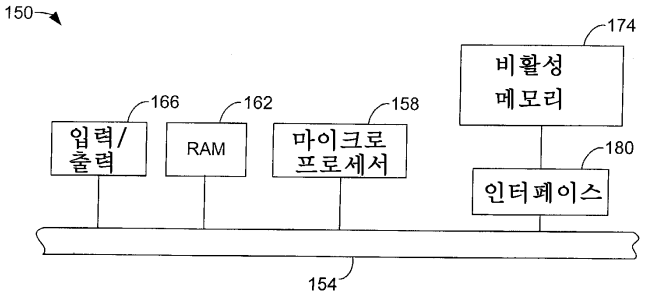
도면3a


100
120
124
비활성 메모리
128
메모리 제어기
116
112
108
입력/출력
RAM
마이크로 프로세서
130
인터페이스
104

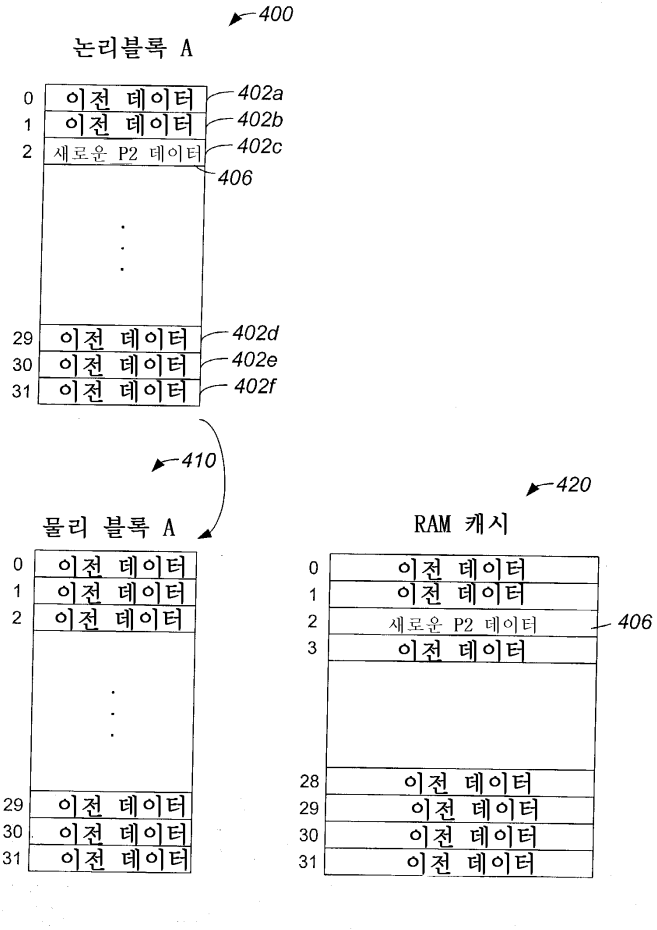
도면3b


120
페이지4
페이지5
페이지6
사용자 데이터
OH
124
11
블록 0
블록 1
...
블록 N
17
주소 디코더
프로그램하고 판독
19
128
21
제어기
23
RAM
ECC
15

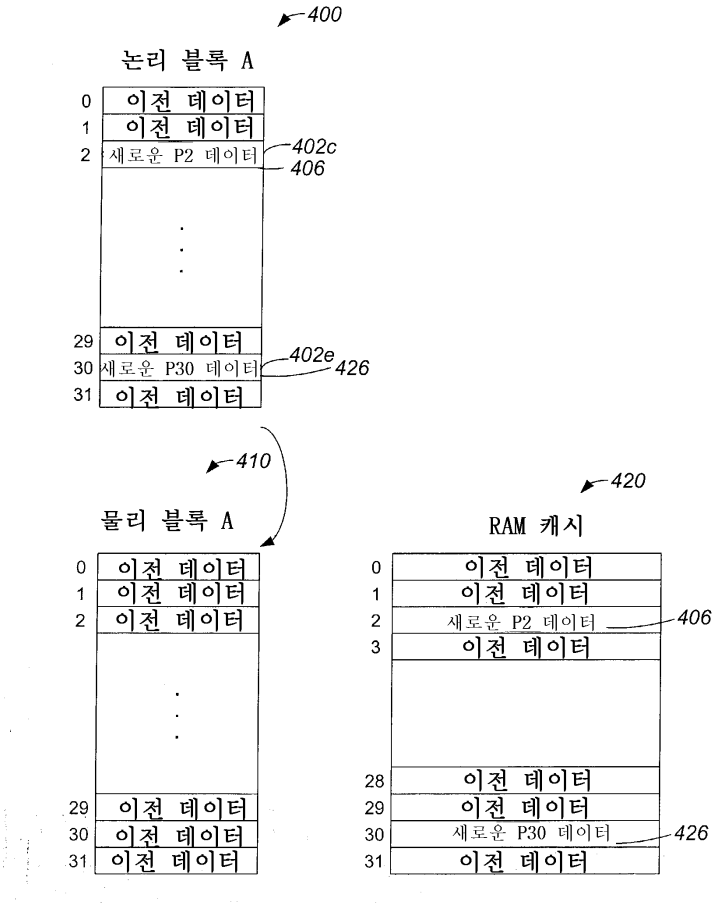
도면3c


150
174
비활성
메모리
166
162
158
입력/
출력
RAM
마이크로
프로세서
180
인터페이스
154

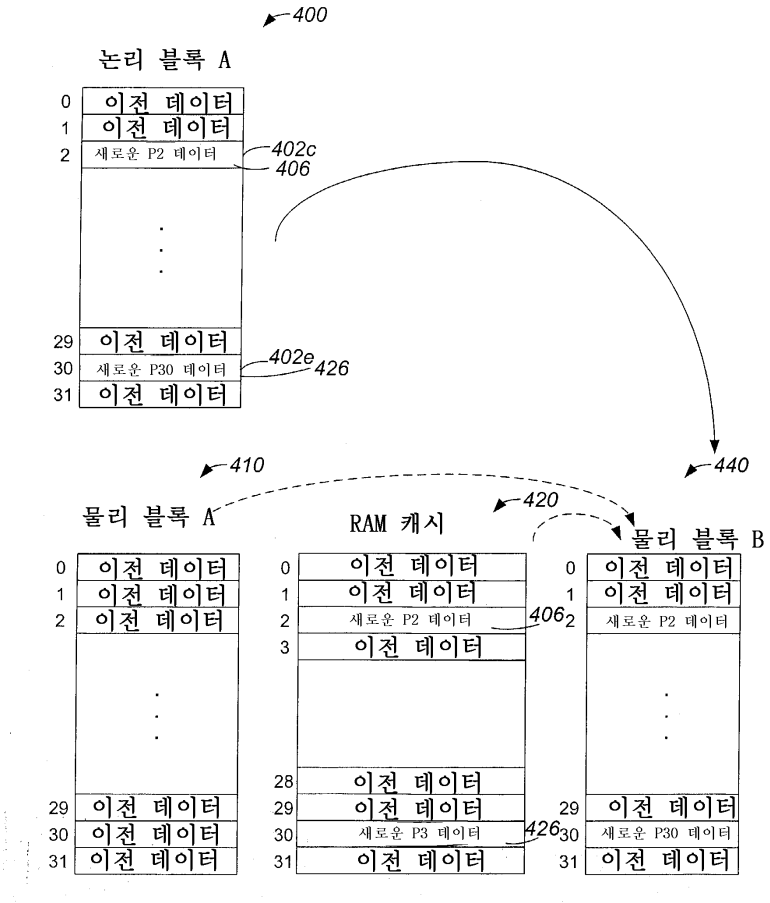
도면4a


400
논리블록 A
0 이전 데이터 402a
1 이전 데이터 402b
2 새로운 P2 데이터 402c
406
29 이전 데이터 402d
30 이전 데이터 402e
31 이전 데이터 402f
410
물리 블록 A
0 이전 데이터
1 이전 데이터
2 이전 데이터
29 이전 데이터
30 이전 데이터
31 이전 데이터
420
RAM 캐시
0 이전 데이터
1 이전 데이터
2 새로운 P2 데이터 406
3 이전 데이터
28 이전 데이터
29 이전 데이터
30 이전 데이터
31 이전 데이터

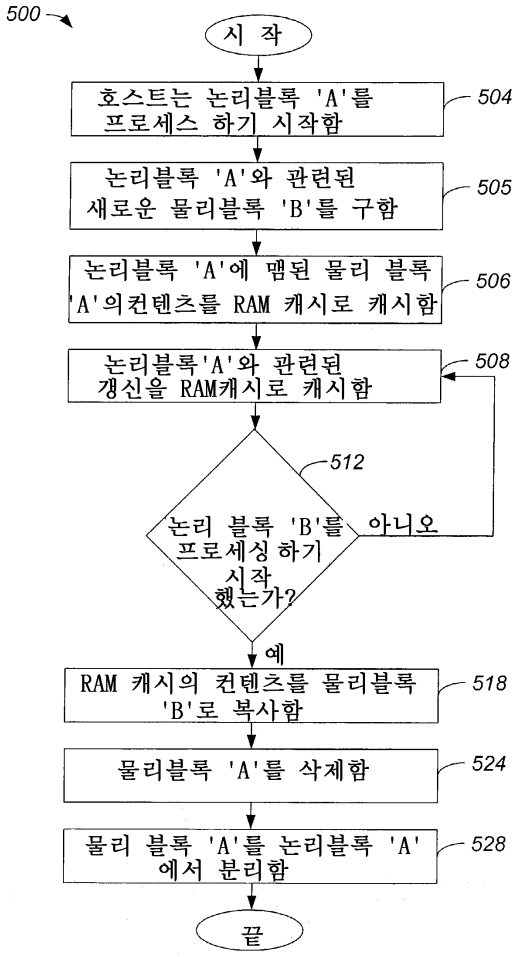
도면4b


400
논리 블록 A
0 이전 데이터
1 이전 데이터
2 새로운 P2 데이터
402c
406
29 이전 데이터
30 새로운 P30 데이터
402e
426
31 이전 데이터
410
물리 블록 A
0 이전 데이터
1 이전 데이터
2 이전 데이터
29 이전 데이터
30 이전 데이터
31 이전 데이터
420
RAM 캐시
0 이전 데이터
1 이전 데이터
2 새로운 P2 데이터
406
3 이전 데이터
28 이전 데이터
29 이전 데이터
30 새로운 P30 데이터
426
31 이전 데이터

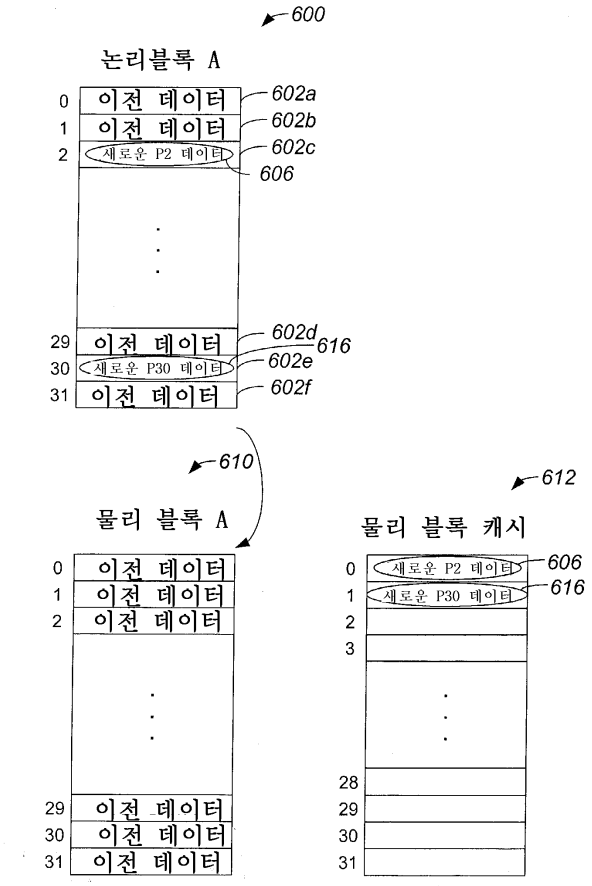
도면4c


400
논리 블록 A
0 이전 데이터
1 이전 데이터
2 새로운 P2 데이터
402c
406
29 이전 데이터
30 새로운 P30 데이터
402e
426
31 이전 데이터
410
440
420
물리 블록 A
RAM 캐시
물리 블록 B
0 이전 데이터
1 이전 데이터
2 이전 데이터
29 이전 데이터
30 이전 데이터
31 이전 데이터
0 이전 데이터
1 이전 데이터
2 새로운 P2 데이터
406
3 이전 데이터
28 이전 데이터
29 이전 데이터
30 새로운 P3 데이터
426
31 이전 데이터
0 이전 데이터
1 이전 데이터
2 새로운 P2 데이터
29 이전 데이터
30 새로운 P30 데이터
31 이전 데이터

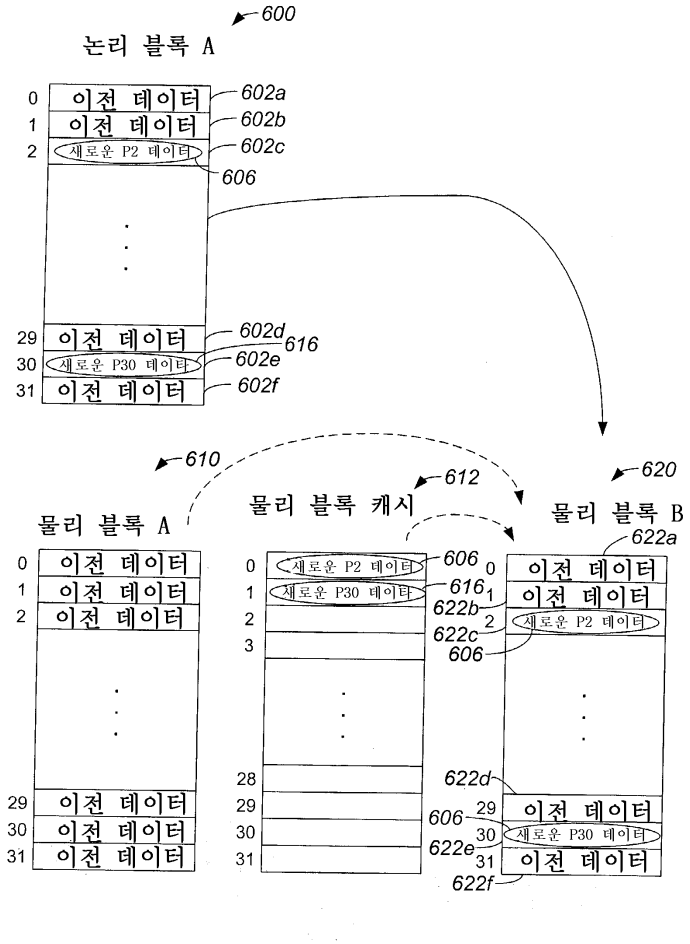
도면5


500
시 작
호스트는 논리블록 'A'를 프로세스 하기 시작함
504
논리블록 'A'와 관련된 새로운 물리블록 'B'를 구함
505
논리블록 'A'에 맵된 물리 블록 'A'의컨텐츠를 RAM 캐시로 캐시함
506
논리블록'A'와 관련된 갱신을 RAM캐시로 캐시함
508
512
논리 블록 'B'를 프로세싱 하기 시작 했는가?
아니오
예
RAM 캐시의 컨텐츠를 물리블록 'B'로 복사함
518
물리블록 'A'를 삭제함
524
물리 블록 'A'를 논리블록 'A'에서 분리함
528
끝

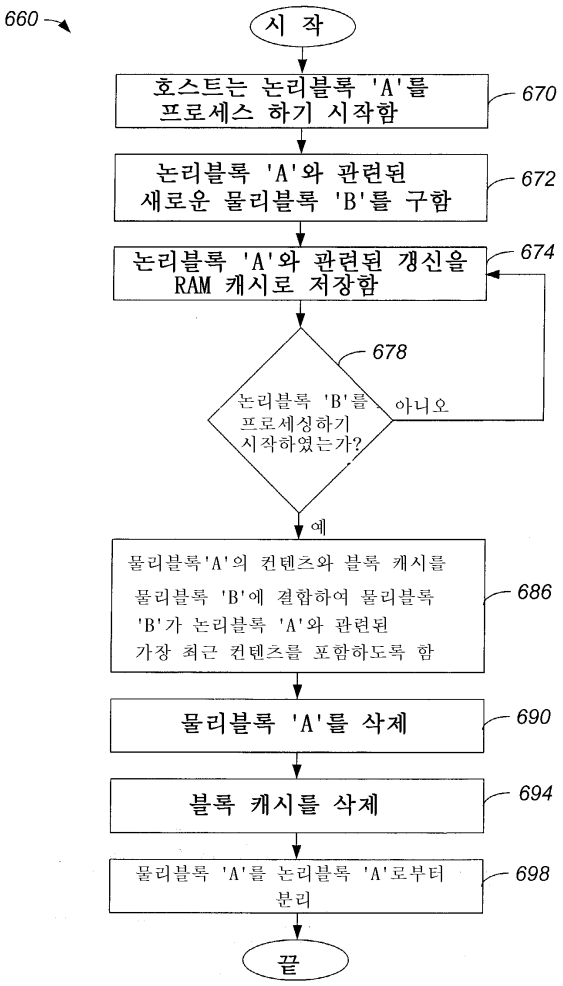
도면6a


600
논리블록 A
0 이전 데이터
1 이전 데이터
2 새로운 P2 데이터
29 이전 데이터
30 새로운 P30 데이터
31 이전 데이터
602a
602b
602c
606
602d
616
602e
602f
610
물리 블록 A
0 이전 데이터
1 이전 데이터
2 이전 데이터
29 이전 데이터
30 이전 데이터
31 이전 데이터
612
물리 블록 캐시
0 새로운 P2 데이터
1 새로운 P30 데이터
2
3
28
29
30
31
606
616

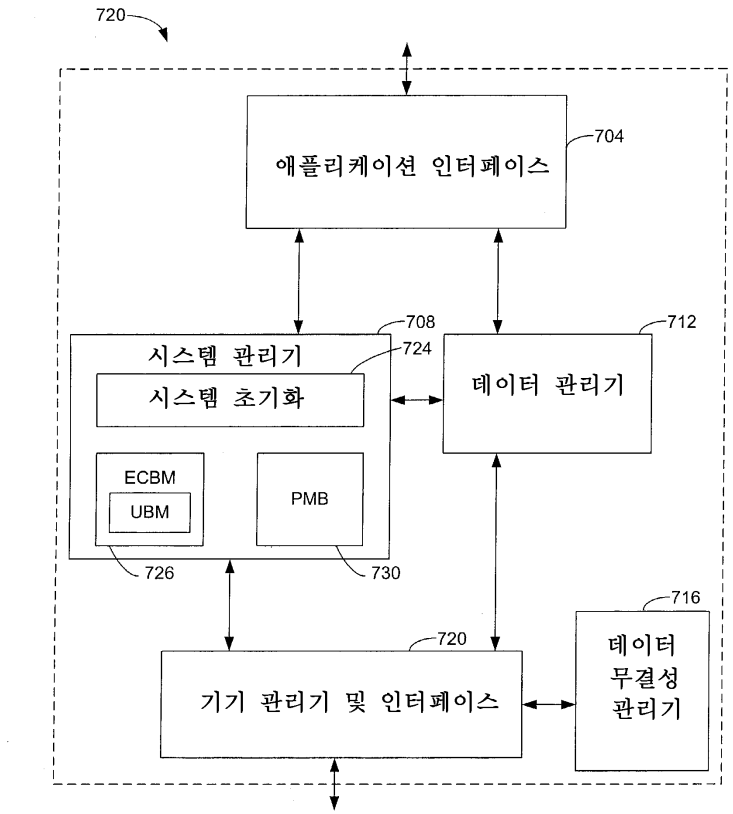
도면6b


600
논리 블록 A
0 이전 데이터 602a
1 이전 데이터 602b
2 새로운 P2 데이터 602c
606
29 이전 데이터 602d
616
30 새로운 P30 데이터 602e
31 이전 데이터 602f
610
612
620
물리 블록 A
물리 블록 캐시
물리 블록 B
622a
622b
622c
622d
622e
622f

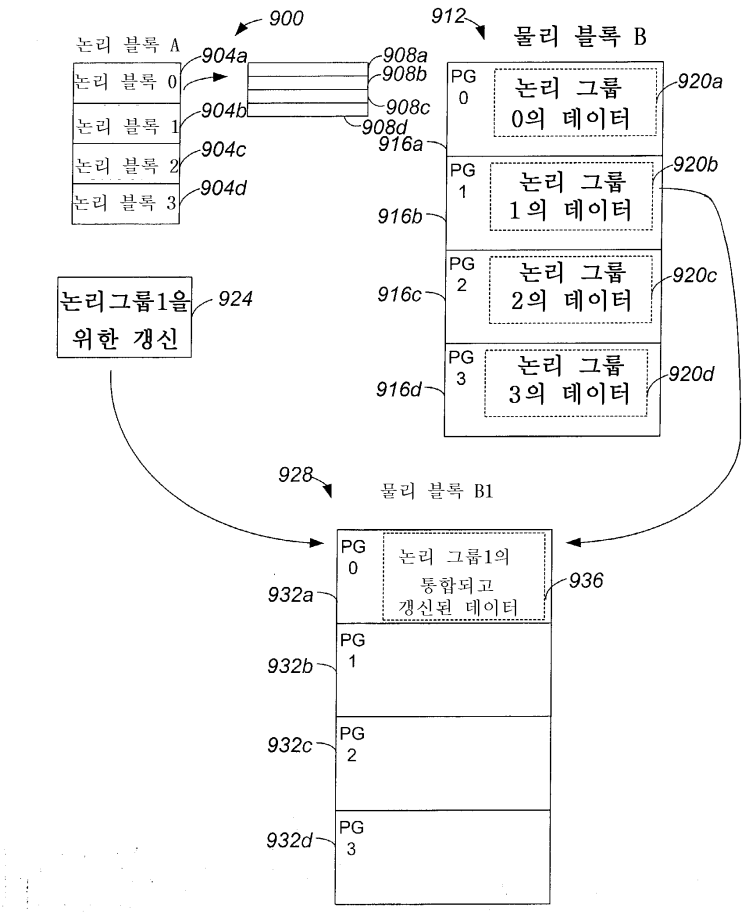
도면7


660
시 작
호스트는 논리블록 'A'를 프로세스 하기 시작함
670
논리블록 'A'와 관련된 새로운 물리블록 'B'를 구함
672
논리블록 'A'와 관련된 갱신을 RAM 캐시로 저장함
674
678
논리블록 'B'를 프로세싱하기 시작하였는가?
아니오
예
물리블록'A'의 컨텐츠와 블록 캐시를 물리블록 'B'에 결합하여 물리블록 'B'가 논리블록 'A'와 관련된 가장 최근 컨텐츠를 포함하도록 함
686
물리블록 'A'를 삭제
690
블록 캐시를 삭제
694
물리블록 'A'를 논리블록 'A'로부터 분리
698
끝

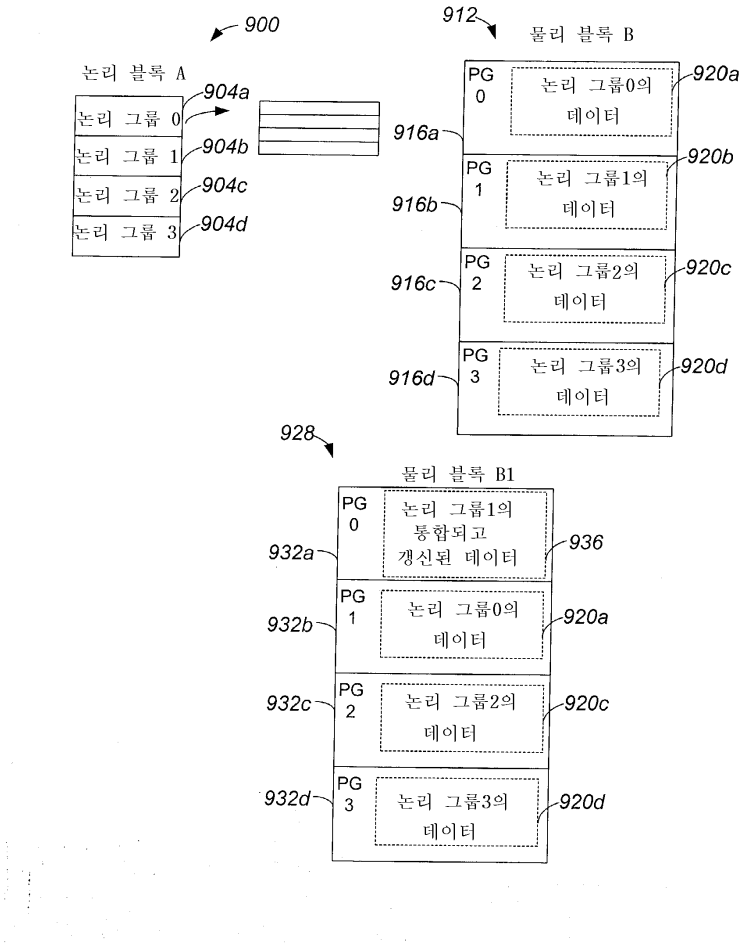
도면8


720
애플리케이션 인터페이스
704
시스템 관리기
708
시스템 초기화
724
ECBM
UBM
PMB
726
730
데이터 관리기
712
716
720
기기 관리기 및 인터페이스
데이터 무결성 관리기

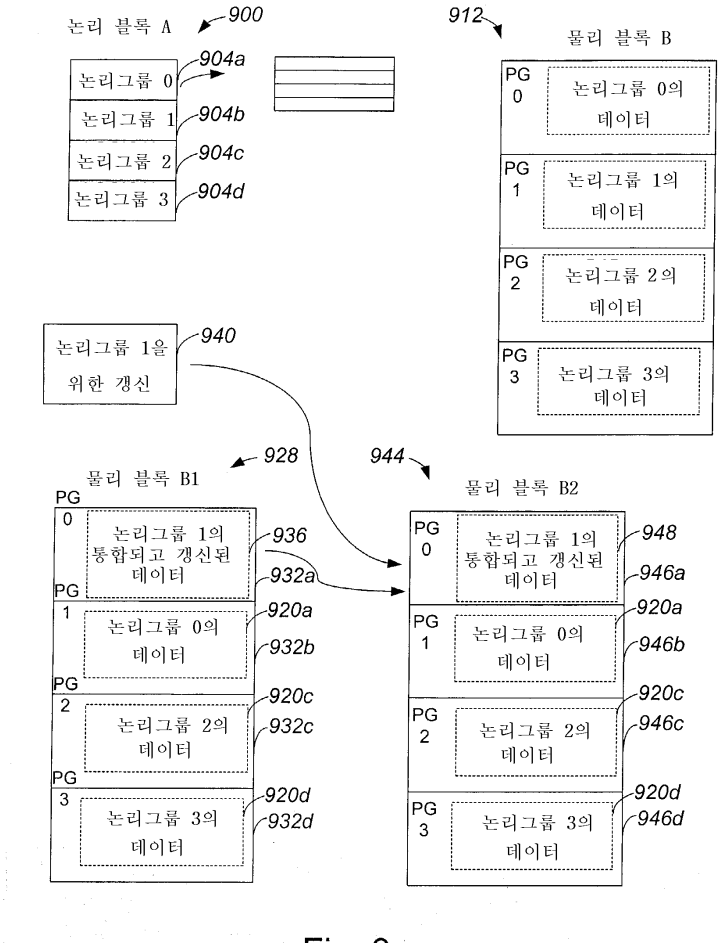
도면9a


900
912
물리 블록 B
논리 블록 A
논리 블록 0
논리 블록 1
논리 블록 2
논리 블록 3
904a
904b
904c
904d
908a
908b
908c
908d
916a
916b
916c
916d
PG 0
PG 1
PG 2
PG 3
논리 그룹 0의 데이터
논리 그룹 1의 데이터
논리 그룹 2의 데이터
논리 그룹 3의 데이터
920a
920b
920c
920d
논리그룹1을 위한 갱신
924
928
물리 블록 B1
논리 그룹1의 통합되고 갱신된 데이터
936
932a
932b
932c
932d

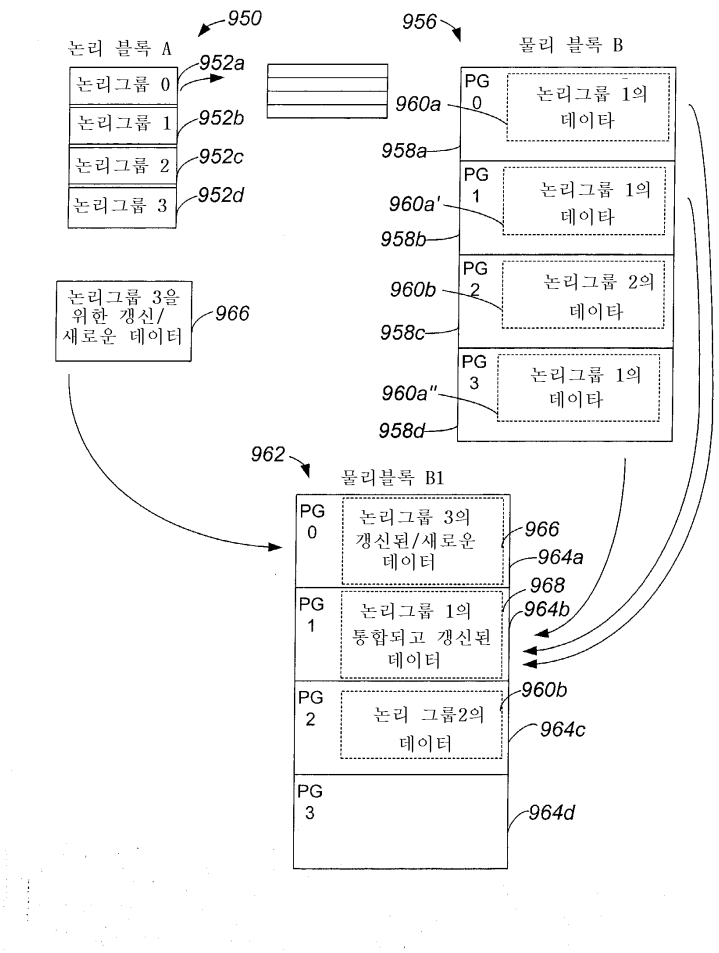
도면9b


900
논리 블록 A
논리 그룹 0
논리 그룹 1
논리 그룹 2
논리 그룹 3
904a
904b
904c
904d
912
물리 블록 B
PG 0
논리 그룹0의 데이터
920a
916a
PG 1
논리 그룹1의 데이터
920b
916b
PG 2
논리 그룹2의 데이터
920c
916c
PG 3
논리 그룹3의 데이터
920d
916d
928
물리 블록 B1
PG 0
논리 그룹1의 통합되고 갱신된 데이터
936
932a
PG 1
논리 그룹0의 데이터
920a
932b
PG 2
논리 그룹2의 데이터
920c
932c
PG 3
논리 그룹3의 데이터
920d
932d

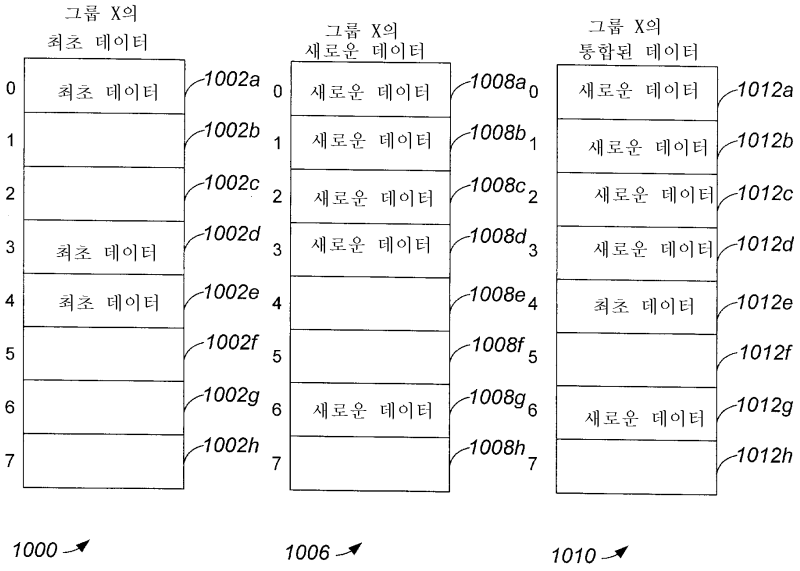
도면9c


논리 블록 A
900
논리그룹 0
904a
논리그룹 1
904b
논리그룹 2
904c
논리그룹 3
904d
912
물리 블록 B
PG 0
논리그룹 0의 데이터
PG 1
논리그룹 1의 데이터
PG 2
논리그룹 2의 데이터
PG 3
논리그룹 3의 데이터
논리그룹 1을 위한 갱신
940
928
물리 블록 B1
PG 0
논리그룹 1의 통합되고 갱신된 데이터
936
932a
PG 1
논리그룹 0의 데이터
920a
932b
PG 2
논리그룹 2의 데이터
920c
932c
PG 3
논리그룹 3의 데이터
920d
932d
944
물리 블록 B2
PG 0
논리그룹 1의 통합되고 갱신된 데이터
948
946a
PG 1
논리그룹 0의 데이터
920a
946b
PG 2
논리그룹 2의 데이터
920c
946c
PG 3
논리그룹 3의 데이터
920d
946d

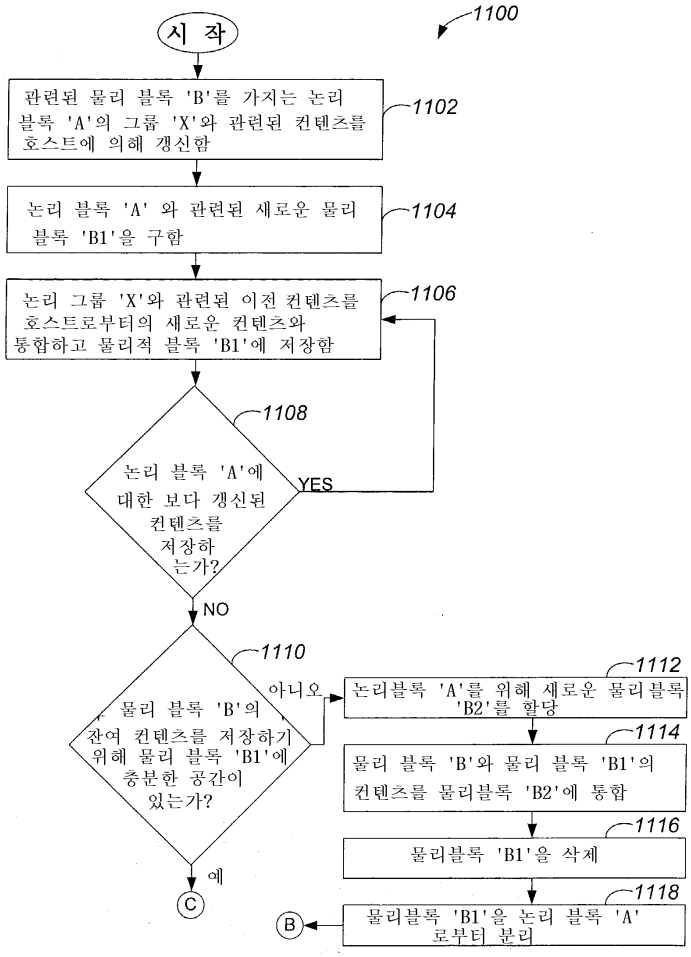
도면9d


950
논리 블록 A
논리그룹 0
논리그룹 1
논리그룹 2
논리그룹 3
952a
952b
952c
952d
956
물리 블록 B
PG 0
논리그룹 1의 데이타
960a
958a
PG 1
논리그룹 1의 데이타
960a'
958b
PG 2
논리그룹 2의 데이타
960b
958c
PG 3
논리그룹 1의 데이타
960a''
958d
논리그룹 3을 위한 갱신/새로운 데이터
966
962
물리블록 B1
PG 0
논리그룹 3의 갱신된/새로운 데이터
966
964a
PG 1
논리그룹 1의 통합되고 갱신된 데이터
968
964b
PG 2
논리 그룹2의 데이터
960b
964c
PG 3
964d

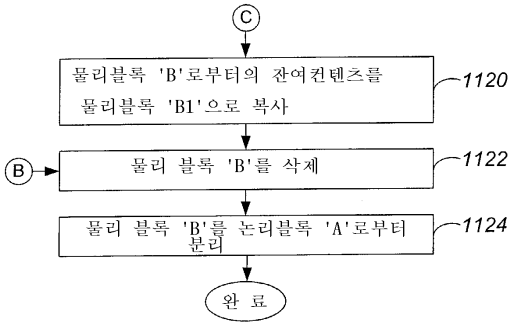
도면10


그룹 X의
최초 데이터
0
최초 데이터
1002a
1
1002b
2
1002c
3
최초 데이터
1002d
4
최초 데이터
1002e
5
1002f
6
1002g
7
1002h
1000
그룹 X의
새로운 데이터
0
새로운 데이터
1008a
1
새로운 데이터
1008b
2
새로운 데이터
1008c
3
새로운 데이터
1008d
4
1008e
5
1008f
6
새로운 데이터
1008g
7
1008h
1006
그룹 X의
통합된 데이터
새로운 데이터
1012a
새로운 데이터
1012b
새로운 데이터
1012c
새로운 데이터
1012d
최초 데이터
1012e
1012f
새로운 데이터
1012g
1012h
1010

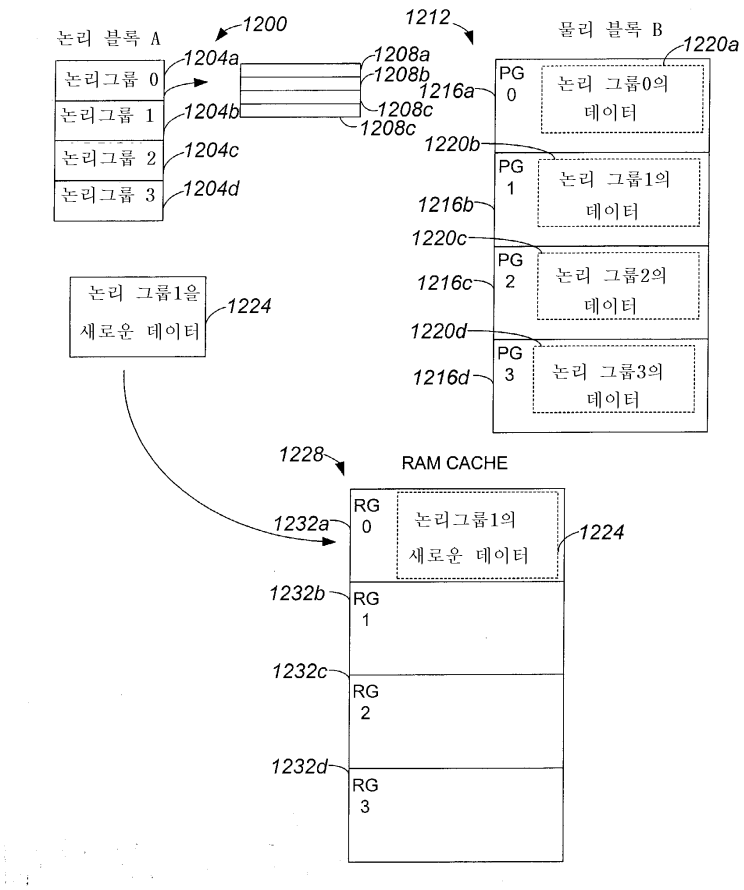
도면11a


1100
시 작
관련된 물리 블록 'B'를 가지는 논리 블록 'A'의 그룹 'X'와 관련된 컨텐츠를 호스트에 의해 갱신함
1102
논리 블록 'A' 와 관련된 새로운 물리 블록 'B1'을 구함
1104
논리 그룹 'X'와 관련된 이전 컨텐츠를 호스트로부터의 새로운 컨텐츠와 통합하고 물리적 블록 'B1'에 저장함
1106
1108
논리 블록 'A'에 대한 보다 갱신된 컨텐츠를 저장하는가?
YES
NO
1110
물리 블록 'B'의 잔여 컨텐츠를 저장하기 위해 물리 블록 'B1'에 충분한 공간이 있는가?
아니오
예
1112
논리블록 'A'를 위해 새로운 물리블록 'B2'를 할당
1114
물리 블록 'B'와 물리 블록 'B1'의 컨텐츠를 물리블록 'B2'에 통합
1116
물리블록 'B1'을 삭제
1118
물리블록 'B1'을 논리 블록 'A'로부터 분리
C
B

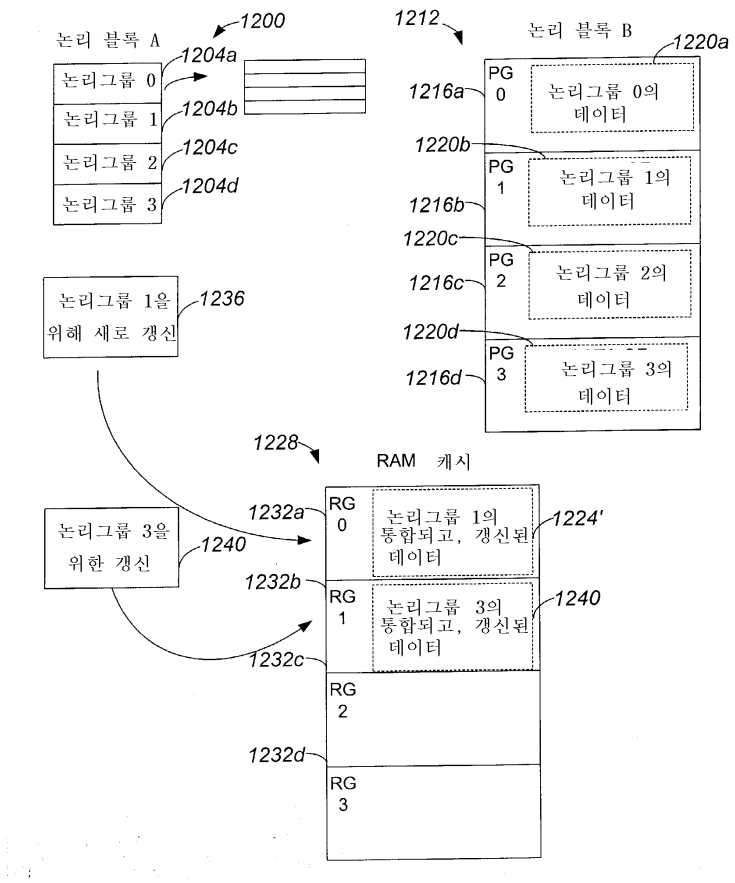
도면11b


C
물리블록 'B'로부터의 잔여컨텐츠를 물리블록 'B1'으로 복사
1120
B
물리 블록 'B'를 삭제
1122
물리 블록 'B'를 논리블록 'A'로부터 분리
1124
완 료

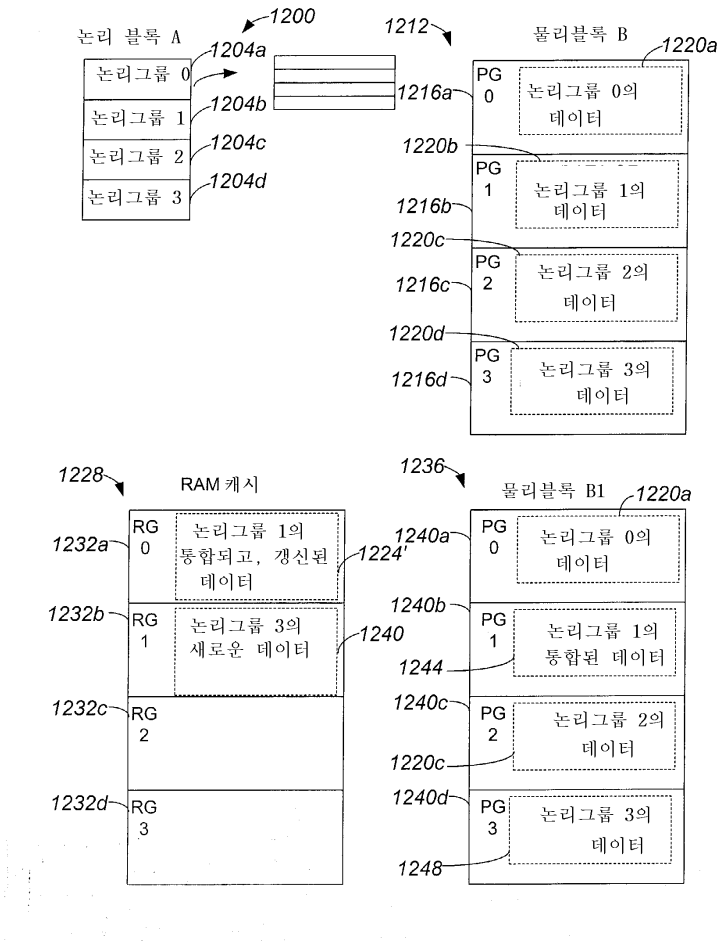
도면12a


논리 블록 A
1200
1212
물리 블록 B
1220a
논리그룹 0
논리그룹 1
논리그룹 2
논리그룹 3
1204a
1204b
1204c
1204d
1208a
1208b
1208c
1208c
1216a
PG 0
논리 그룹0의 데이터
1220b
PG 1
논리 그룹1의 데이터
1216b
1220c
PG 2
논리 그룹2의 데이터
1216c
1220d
PG 3
논리 그룹3의 데이터
1216d
논리 그룹1을 새로운 데이터
1224
1228
RAM CACHE
RG 0
논리그룹1의 새로운 데이터
1224
1232a
1232b
RG 1
1232c
RG 2
1232d
RG 3

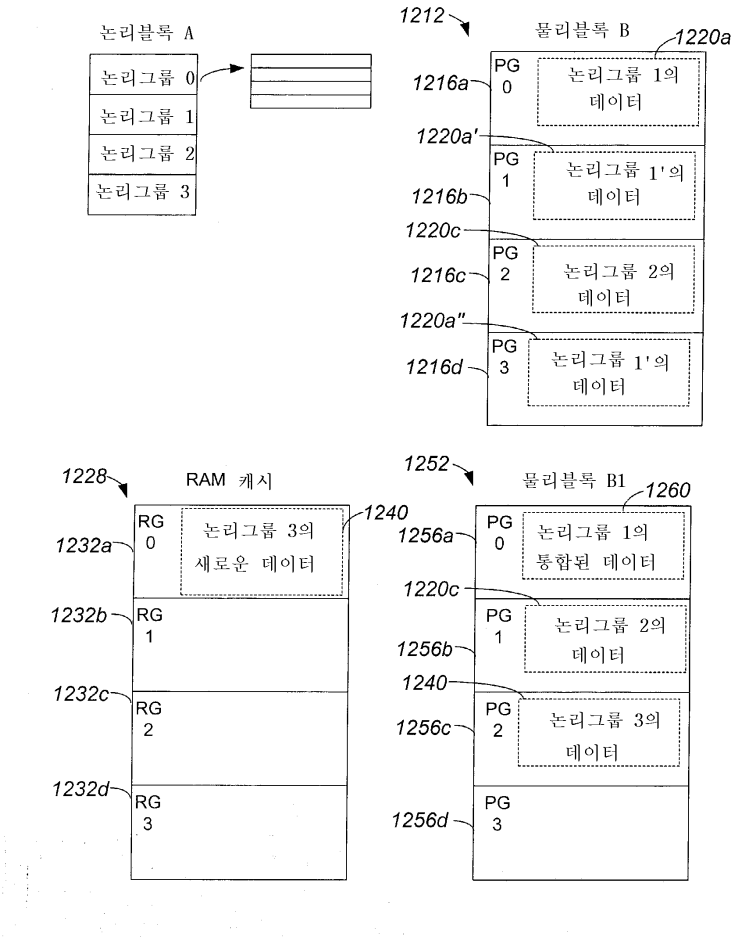
도면12b


논리 블록 A
1200
1204a
논리그룹 0
1204b
논리그룹 1
1204c
논리그룹 2
1204d
논리그룹 3
1212
논리 블록 B
1220a
PG 0
1216a
논리그룹 0의 데이터
1220b
PG 1
1216b
논리그룹 1의 데이터
1220c
PG 2
1216c
논리그룹 2의 데이터
1220d
PG 3
1216d
논리그룹 3의 데이터
논리그룹 1을 위해 새로 갱신
1236
1228
RAM 캐시
1232a
RG 0
논리그룹 1의 통합되고, 갱신된 데이터
1224'
논리그룹 3을 위한 갱신
1240
1232b
RG 1
논리그룹 3의 통합되고, 갱신된 데이터
1240
1232c
RG 2
1232d
RG 3

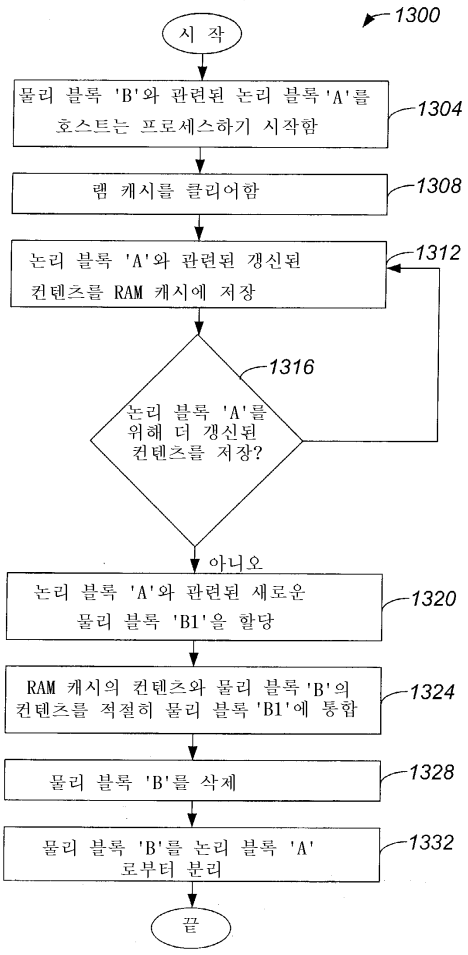
도면12c


논리 블록 A
1200
1204a
1204b
1204c
1204d
논리그룹 0
논리그룹 1
논리그룹 2
논리그룹 3
1212
물리블록 B
1220a
1216a
1220b
1216b
1220c
1216c
1220d
1216d
PG 0
PG 1
PG 2
PG 3
논리그룹 0의 데이터
논리그룹 1의 데이터
논리그룹 2의 데이터
논리그룹 3의 데이터
1228
RAM 캐시
1232a
1232b
1232c
1232d
RG 0
RG 1
RG 2
RG 3
논리그룹 1의 통합되고, 갱신된 데이터
논리그룹 3의 새로운 데이터
1224'
1240
1236
물리블록 B1
1220a
1240a
1240b
1244
1240c
1220c
1240d
1248
PG 0
PG 1
PG 2
PG 3
논리그룹 0의 데이터
논리그룹 1의 통합된 데이터
논리그룹 2의 데이터
논리그룹 3의 데이터

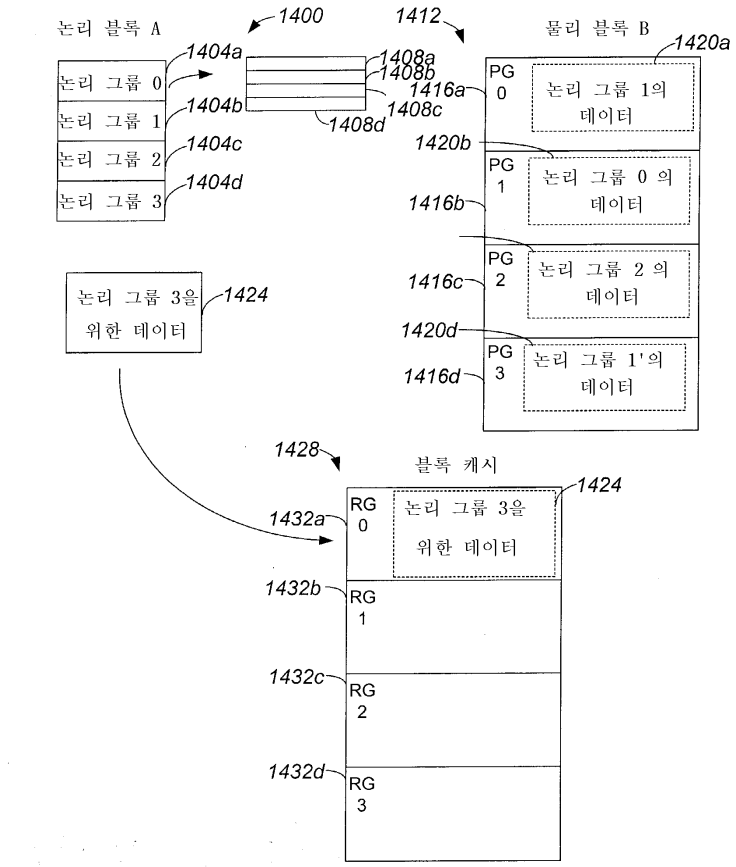
도면12d


논리블록 A
논리그룹 0
논리그룹 1
논리그룹 2
논리그룹 3
1212
물리블록 B
1220a
1216a
PG 0
논리그룹 1의 데이터
1220a'
PG 1
논리그룹 1'의 데이터
1216b
1220c
PG 2
논리그룹 2의 데이터
1216c
1220a''
PG 3
논리그룹 1'의 데이터
1216d
1228
RAM 캐시
RG 0
논리그룹 3의 새로운 데이터
1240
1232a
RG 1
1232b
RG 2
1232c
RG 3
1232d
1252
물리블록 B1
1260
1256a
PG 0
논리그룹 1의 통합된 데이터
1220c
PG 1
논리그룹 2의 데이터
1256b
1240
PG 2
논리그룹 3의 데이터
1256c
PG 3
1256d

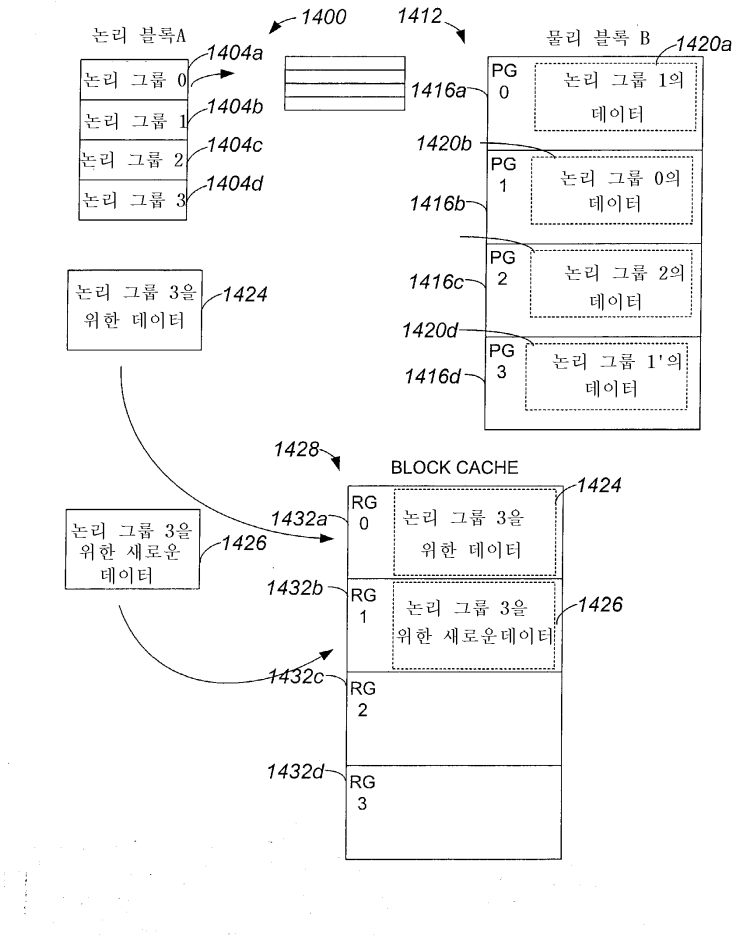
도면13


1300
시 작
물리 블록 'B'와 관련된 논리 블록 'A'를 호스트는 프로세스하기 시작함
1304
램 캐시를 클리어함
1308
논리 블록 'A'와 관련된 갱신된 컨텐츠를 RAM 캐시에 저장
1312
1316
논리 블록 'A'를 위해 더 갱신된 컨텐츠를 저장?
아니오
논리 블록 'A'와 관련된 새로운 물리 블록 'B1'을 할당
1320
RAM 캐시의 컨텐츠와 물리 블록 'B'의 컨텐츠를 적절히 물리 블록 'B1'에 통합
1324
물리 블록 'B'를 삭제
1328
물리 블록 'B'를 논리 블록 'A'로부터 분리
1332
끝

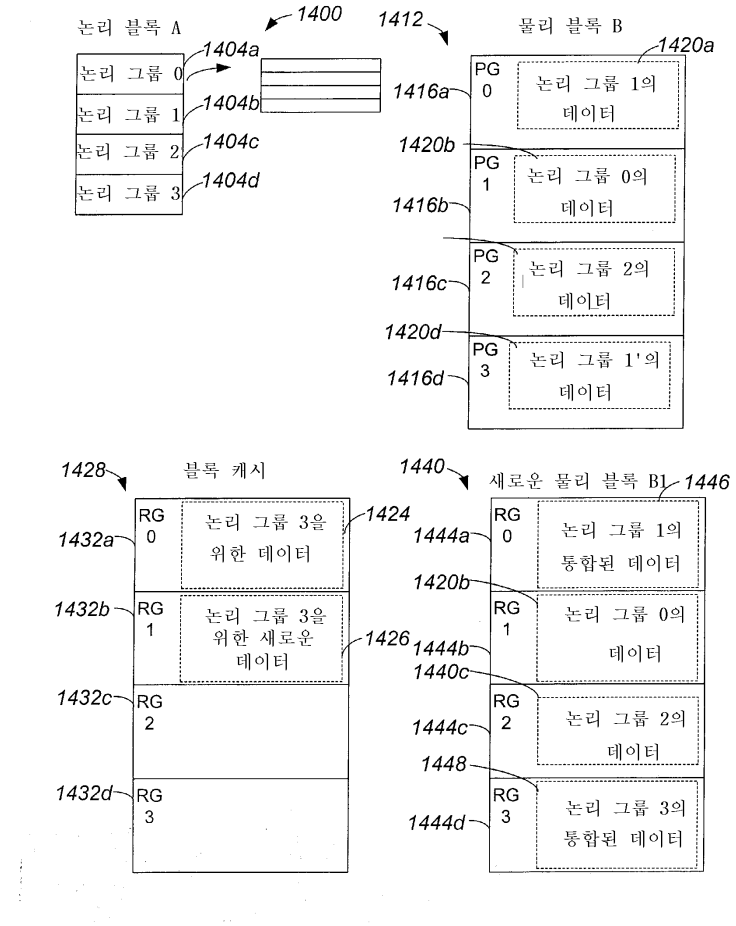
도면14a


논리 블록 A
1400
1412
물리 블록 B
1420a
논리 그룹 0
논리 그룹 1
논리 그룹 2
논리 그룹 3
1404a
1404b
1404c
1404d
1408a
1408b
1408c
1408d
1416a
PG 0
논리 그룹 1의 데이터
1420b
PG 1
논리 그룹 0 의 데이터
1416b
PG 2
논리 그룹 2 의 데이터
1416c
1420d
PG 3
논리 그룹 1'의 데이터
1416d
논리 그룹 3을 위한 데이터
1424
1428
블록 캐시
RG 0
논리 그룹 3을 위한 데이터
1432a
1432b
RG 1
1432c
RG 2
1432d
RG 3

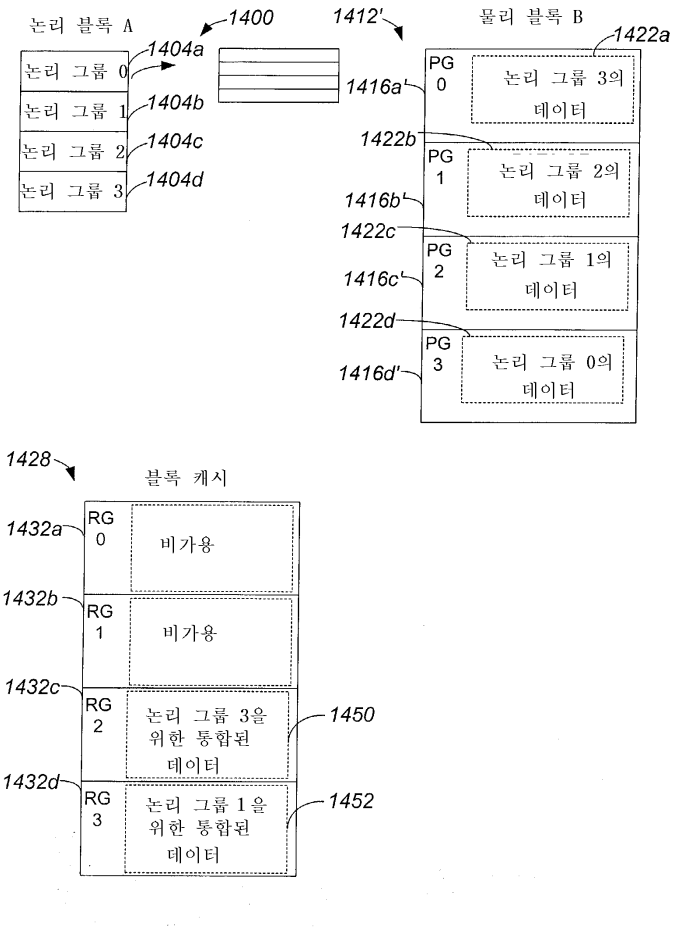
도면14b


1400
1412
논리 블록A
1404a
논리 그룹 0
1404b
논리 그룹 1
1404c
논리 그룹 2
1404d
논리 그룹 3
물리 블록 B
1420a
PG 0
논리 그룹 1의 데이터
1416a
1420b
PG 1
논리 그룹 0의 데이터
1416b
PG 2
논리 그룹 2의 데이터
1416c
1420d
PG 3
논리 그룹 1'의 데이터
1416d
논리 그룹 3을 위한 데이터
1424
1428
BLOCK CACHE
1424
RG 0
논리 그룹 3을 위한 데이터
1432a
논리 그룹 3을 위한 새로운 데이터
1426
1432b
RG 1
논리 그룹 3을 위한 새로운데이터
1426
1432c
RG 2
1432d
RG 3

도면14c


논리 블록 A
1400
1412
물리 블록 B
1404a
1420a
논리 그룹 0
논리 그룹 1
논리 그룹 2
논리 그룹 3
1404b
1404c
1404d
1416a
PG 0
논리 그룹 1의 데이터
1420b
PG 1
논리 그룹 0의 데이터
1416b
PG 2
논리 그룹 2의 데이터
1416c
1420d
PG 3
논리 그룹 1'의 데이터
1416d
1428
블록 캐시
1440
새로운 물리 블록 B1
1446
RG 0
논리 그룹 3을 위한 데이터
1424
1432a
RG 1
논리 그룹 3을 위한 새로운 데이터
1426
1432b
RG 2
1432c
RG 3
1432d
1444a
RG 0
논리 그룹 1의 통합된 데이터
1420b
RG 1
논리 그룹 0의 데이터
1444b
1440c
RG 2
논리 그룹 2의 데이터
1444c
1448
RG 3
논리 그룹 3의 통합된 데이터
1444d

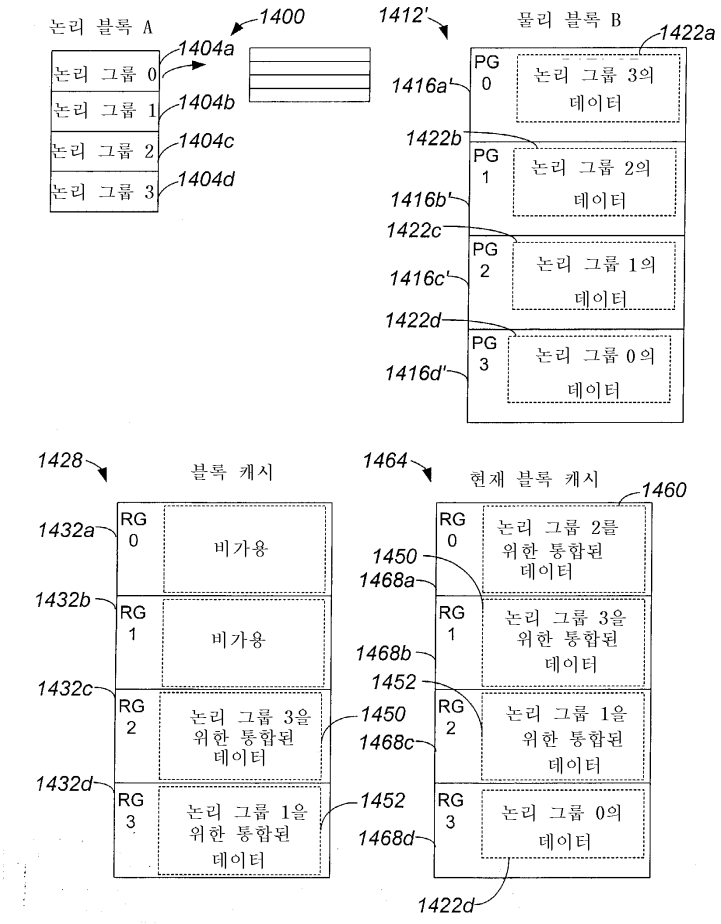
도면14d



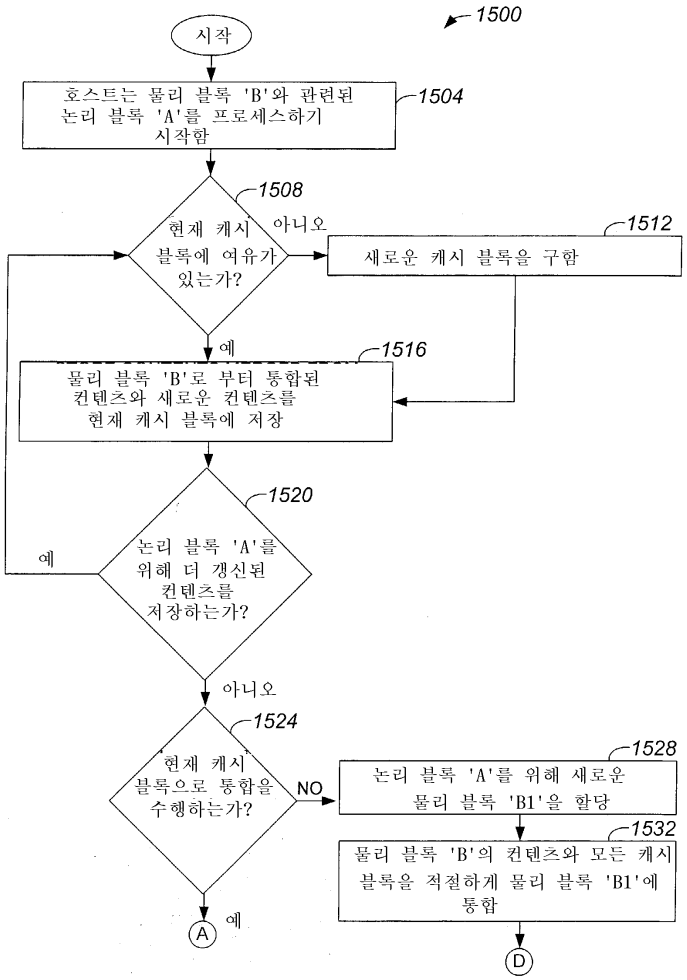
도면14e

삭제

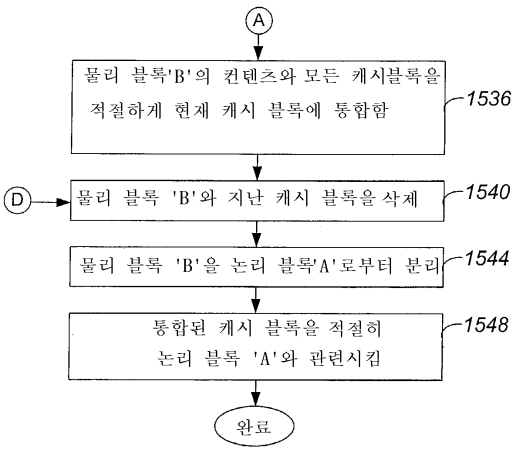
도면14f


논리 블록 A
1400
1404a
논리 그룹 0
1404b
논리 그룹 1
1404c
논리 그룹 2
1404d
논리 그룹 3
1412'
물리 블록 B
1422a
1416a'
PG 0
논리 그룹 3의 데이터
1422b
PG 1
논리 그룹 2의 데이터
1416b'
1422c
PG 2
논리 그룹 1의 데이터
1416c'
1422d
PG 3
논리 그룹 0의 데이터
1416d'
1428
블록 캐시
1432a
RG 0
비가용
1432b
RG 1
비가용
1432c
RG 2
논리 그룹 3을 위한 통합된 데이터
1450
1432d
RG 3
논리 그룹 1을 위한 통합된 데이터
1452
1464
현재 블록 캐시
1460
RG 0
논리 그룹 2를 위한 통합된 데이터
1450
1468a
RG 1
논리 그룹 3을 위한 통합된 데이터
1468b
1452
RG 2
논리 그룹 1을 위한 통합된 데이터
1450
1468c
RG 3
논리 그룹 0의 데이터
1452
1468d
1422d

도면15a


1500
시작
호스트는 물리 블록 'B'와 관련된 논리 블록 'A'를 프로세스하기 시작함
1504
1508
현재 캐시 블록에 여유가 있는가?
아니오
1512
새로운 캐시 블록을 구함
예
1516
물리 블록 'B'로 부터 통합된 컨텐츠와 새로운 컨텐츠를 현재 캐시 블록에 저장
1520
논리 블록 'A'를 위해 더 갱신된 컨텐츠를 저장하는가?
예
아니오
1524
현재 캐시 블록으로 통합을 수행하는가?
NO
1528
논리 블록 'A'를 위해 새로운 물리 블록 'B1'을 할당
1532
물리 블록 'B'의 컨텐츠와 모든 캐시 블록을 적절하게 물리 블록 'B1'에 통합
A
예
D

도면15b


A
물리 블록'B'의 컨텐츠와 모든 캐시블록을 적절하게 현재 캐시 블록에 통합함
1536
D
물리 블록 'B'와 지난 캐시 블록을 삭제
1540
물리 블록 'B'을 논리 블록'A'로부터 분리
1544
통합된 캐시 블록을 적절히 논리 블록 'A'와 관련시킴
1548
완료