



(19)대한민국특허청(KR)  
(12) 공개특허공보(A)

(51) 。 Int. Cl.

G06F 12/00 (2006.01)

G06F 12/02 (2006.01)

(11) 공개번호 10-2007-0060070

(43) 공개일자 2007년06월12일

(21) 출원번호 10-2007-7001795

(22) 출원일자 2007년01월24일

심사청구일자 없음

번역문 제출일자 2007년01월24일

(86) 국제출원번호 PCT/US2005/021846

(87) 국제공개번호 WO 2006/019496

국제출원일자 2005년06월17일

국제공개일자 2006년02월23일

(30) 우선권주장 10/897,049 2004년07월21일 미국(US)  
11/022,369 2004년12월23일 미국(US)

(71) 출원인 쉐디스크 코퍼레이션  
미합중국, 캘리포니아주 95035, 밀피타스, 맥카시 볼레바드 601

(72) 발명자 콘레이, 케빈 엠.  
미국, 캘리포니아 95120, 산 호세, 맨쿠소 스트리트 6152  
신클레어, 알란 웰시  
영국, 매디스톤 팔키르크 에프케이2 0비유, 캔디,브로드헤드, 더 코티지  
스  
스미스, 피터 존  
영국, 미들로시안 스코트랜드 이에이치22 3에이치에이,이스크뱅크, 보  
나리그 로드 21

(74) 대리인 박경재  
송범엽

전체 청구항 수 : 총 21 항

## (54) 최적화된 순차적인 클러스터 관리용 애플리케이션 분석

### (57) 요약

비-휘발성 메모리 시스템에서(예를 들어, 플래시 메모리) 데이터를 관리하는 기술이 개시된다. 제어기는 호스트의 파일 시스템에 관한 정보를 이용할 수 있고, 하나 이상의 클러스터(또는 클러스터를 구비한 섹터)가 현재 할당되는지의 여부를 결정하도록 이것은 비-휘발성 메모리 위에서 호스트에 의해 저장된다. 상기 제어기는 호스트의 파일 시스템에 관한 정보를 이용하여, 언제 호스트가 다음의 자유(free) 클러스터로 데이터를 보내어 비-휘발성 메모리의 다른 위치로부터 데이터를 복사하여 순차적인 형식으로 그러한 데이터를 저장하는 지를 식별한다.

### 대표도

도 2a

## 특허청구의 범위

### 청구항 1.

제어기와 비-휘발성 메모리 어레이를 포함하는 메모리 시스템에서, 적응형(adaptive) 방식으로 데이터의 부분을 저장하는 방법에 있어서,

비-휘발성 메모리 어레이 내에서 디렉토리(directory) 혹은 파일 할당 테이블(File Allocation Table)에서 하나의 호스트에 의해 저장된 정보로부터 논리 어드레스 범위의 애트리뷰트(attribute)를 상기 제어기가 얻고;

논리 어드레스 범위의 애트리뷰트에 따라 논리 어드레스 범위내에 논리 어드레스를 갖는 데이터의 부분에 대해 복수의 저장 체계(storage schemes)로부터 저장 체계를 제어기가 선택하고; 및

상기 선택된 저장 체계에 따라서 데이터의 부분을 제어기가 저장하는 것을 특징으로 하는, 적응형 방식으로 데이터의 부분을 저장하는 방법.

### 청구항 2.

제 1 항에 있어서, 상기 논리 어드레스 범위의 애트리뷰트는 상기 논리 어드레스 범위가 호스트 데이터의 저장을 위해 다음의 자유(free) 논리 어드레스 범위인 것을 특징으로 하는, 적응형 방식으로 데이터의 부분을 저장하는 방법.

### 청구항 3.

제 2 항에 있어서, 상기 선택된 저장 체계는 순차적인 갱신 블록에서 데이터의 부분의 저장을 포함하는, 적응형 방식으로 데이터의 부분을 저장하는 방법.

### 청구항 4.

제 1 항에 있어서, 상기 논리 어드레스 범위의 애트리뷰트는 논리 어드레스 범위가 호스트 데이터의 저장을 위해 그 다음의 자유 논리 어드레스 범위가 아닌 것을 특징으로 하는, 적응형 방식으로 데이터의 부분을 저장하는 방법.

### 청구항 5.

제4항에 있어서, 상기 선택된 저장 체계가 카오틱(chaotic) 갱신 블록에서 데이터의 부분의 저장을 포함하는, 적응형 방식으로 데이터의 부분을 저장하는 방법.

### 청구항 6.

제 1 항에 있어서, 데이터의 부분이 비-휘발성 메모리 어레이의 제1 위치에서 이전에 저장되고, 상기 논리 어드레스 범위의 애트리뷰트는 상기 논리 어드레스 범위가 호스트에 의해 할당되고 상기 선택된 저장 체계는 비-휘발성 메모리 어레이에서 제1 위치에서 제2 위치로 상기 데이터의 부분의 복사를 포함하는 것을 특징으로 하는, 적응형 방식으로 데이터의 부분을 저장하는 방법.

### 청구항 7.

제 6항에 있어서, 상기 데이터의 부분이 비-휘발성 메모리 어레이의 제1 위치에 이전에 저장되고, 논리 어드레스 범위의 애트리뷰트는 논리 어드레스 범위가 호스트에 의해 해제되고 상기 선택된 저장 체계는 제1 위치에서 제2 위치까지 데이터의 부분의 복사없이 상기 선택된 저장 체계가 제1 위치에서 데이터의 부분을 유지하는 것을 특징으로 하는, 적응형 방식으로 데이터의 부분을 저장하는 방법.

## 청구항 8.

제1항에 있어서, 파일 할당 테이블 또는 디렉토리로부터 유도되는 데이터 애트리뷰트 저장으로부터 얻어진 상기 애트리뷰트인 것을 특징으로 하는, 적응형 방식으로 데이터의 부분을 저장하는 방법.

## 청구항 9.

비-휘발성 메모리 어레이에서 호스트로부터 수신된 데이터 저장 방법에 있어서,

상기 호스트로부터 하나 이상의 어드레스 가능한 유닛을 수신하고;

데이터의 저장을 위해 호스트에 가용한 복수의 논리 어드레스 범위를 식별하도록 상기 비-휘발성 메모리에서 호스트에 의해 저장된 정보를 분석하고;

하나 이상의 어드레스 가능한 데이터 유닛이 상기 호스트에 가용한 그 다음 범위의 논리 어드레스에 할당되는지를 결정하도록 호스트에 가용한 복수의 논리 어드레스 범위와 하나 이상의 어드레스 가능한 데이터의 유닛의 논리 어드레스를 비교하고; 및

결정에 좌우되는 방식으로 하나 이상의 어드레스 가능한 데이터 유닛을 저장하는 단계를 포함하는, 비-휘발성 메모리 어레이에서 호스트로부터 수신된 데이터 저장 방법.

## 청구항 10.

제 9항에 있어서, 하나 이상의 어드레스 가능한 유닛이 다음의 가용한 순차적인 위치에 할당된다면 하나 이상의 어드레스 가능한 유닛의 저장 방식은 순차적인 방식이고, 한 이상의 어드레스 가능한 유닛이 다음의 가용한 순차적인 위치에 할당되지 않는다면 저장의 방식은 비-순차적인 방식인, 비-휘발성 메모리 어레이에서 호스트로부터 수신된 데이터 저장 방법.

## 청구항 11.

제 9항에 있어서, 하나 이상의 어드레스 가능한 데이터 유닛의 논리 어드레스와 마지막으로 기록된 어드레스 가능한 데이터 유닛의 논리 어드레스 사이의 논리 어드레스 갭(address gap)이 소정의 양을 초과하는지의 여부를 포함하여 부가적인 요소에 좌우되는 방식으로 하나 이상의 어드레스 가능한 데이터 유닛을 저장하는, 비-휘발성 메모리 어레이에서 호스트로부터 수신된 데이터 저장 방법.

## 청구항 12.

제 11항에 있어서, 하나 이상의 어드레스 가능한 데이터 유닛을 저장하는 갱신 블록이 순차적인 형태로 소정 양의 데이터 보다 더 많이 포함하는지의 여부가 부가적인 요소인, 비-휘발성 메모리 어레이에서 호스트로부터 수신된 데이터 저장 방법.

## 청구항 13.

어드레스 가능한 데이터 유닛에서 호스트로부터 수신되는 데이터를 비-휘발성 메모리에 기억하는 메모리 장치에 있어서,  
호스트 파일 할당 테이블의 복사를 포함하는 비-휘발성 메모리 어레이; 및  
비-휘발성 메모리 어레이와 통신하는 메모리 제어기를 포함하며,

상기 메모리 제어기가 호스트 파일 할당 테이블의 복사의 내용을 모니터하고, 상기 메모리 제어기가 상기 호스트 파일 할당 테이블의 복사로부터 논리 어드레스 공간의 부분의 할당 상태를 결정하고, 상기 메모리 제어기가 할당 상태에 응답하여 논리 어드레스 공간의 부분 내에 논리 어드레스를 갖는 어드레스 가능한 데이터 유닛에 대해 데이터 관리 체계를 선택하는 것을 특징으로 하는, 비-휘발성 메모리에서 데이터를 저장하는 메모리 장치.

#### 청구항 14.

제 13항에 있어서, 상기 메모리 장치는 제어기에 분리 가능하게 접속된 분리 가능한 메모리 카드의 일부를 형성하는 것을 특징으로 하는, 비-휘발성 메모리에서 데이터를 저장하는 메모리 장치.

#### 청구항 15.

제 13항에 있어서, 상기 메모리 제어기는 데이터 애트리뷰트 저장(Data Attribute Store)에 논리 어드레스 공간의 부분의 할당 상태를 기록하는 것을 특징으로 하는, 비-휘발성 메모리에서 데이터를 저장하는 메모리 장치.

#### 청구항 16.

제13항에 있어서, 어드레스 가능한 데이터 유닛이 비-휘발성 메모리에 저장을 위해 호스트로부터 수신되는, 비-휘발성 메모리에서 데이터를 저장하는 메모리 장치.

#### 청구항 17.

제 13항에 있어서, 메모리 장치가 가비지(garbage) 수집 동작을 실행할 때, 상기 어드레스 가능한 데이터 유닛이 비-휘발성 메모리 어레이의 제1 위치에 저장되고, 논리 어드레스 공간의 부분이 할당된다면 데이터 관리 체계가 비-휘발성 메모리 어레이의 제2 위치에 어드레스 가능한 데이터 유닛을 복사하고, 논리 어드레스 공간의 부분이 할당되지 않으면 데이터 관리 체계가 제1 위치에서 어드레스 가능한 데이터 유닛을 유지하는, 비-휘발성 메모리에서 데이터를 저장하는 메모리 장치.

#### 청구항 18.

비-휘발성 메모리에서 호스트 데이터의 저장을 위해 메모리 제어기에 가용한 대안의 저장 체계 사이에 선택하기 위한 프로그램 코드를 포함하는 컴퓨터 판독 가능 매체에 있어서,

비-휘발성 메모리에서 호스트에 의해 저장된 정보로부터 논리 어드레스 공간의 복수의 부분의 할당 상태를 식별하여 메모리 제어기에 상기 할당 상태를 제공하는 컴퓨터 프로그램 코드; 및

데이터의 상기 부분의 논리 어드레스를 포함하는 일부의 논리 어드레스 공간의 할당 상태에 따라 호스트 데이터의 부분에 대해 데이터 관리 체계를 선택하는 컴퓨터 프로그램 코드를 포함하는, 프로그램 코드를 포함하는 컴퓨터 판독 가능 매체.

#### 청구항 19.

제 18항에 있어서, 논리 어드레스 공간의 부분이 할당되지 않은 논리 어드레스 공간의 다음 부분이고 달리 케이오틱 블록에 데이터의 부분을 저장한다면, 순차적인 블록에 데이터의 부분을 저장하도록 선택된 관리 체계에 따라 호스트 데이터의 부분에서 동작하는 컴퓨터 프로그램 코드를 더 포함하는, 컴퓨터 판독 가능 매체.

## 청구항 20.

제 18항에 있어서, 논리 어드레스 공간의 부분이 할당되지 않으면 새로운 위치로 복사함이 없이 지난 위치에 데이터의 부분을 남기고, 논리 어드레스 공간의 부분이 할당되면 지난 위치에서 새로운 위치로 데이터의 부분을 복사하도록 선택된 관리 체계에 따라 호스트 데이터의 부분에서 동작하는 컴퓨터 프로그램 코드를 더 포함하는 컴퓨터 판독 가능 매체

## 청구항 21.

제 18항에 있어서, 논리 어드레스 공간의 복수의 부분의 할당 상태를 포함하는 데이터 애트리뷰트 저장(Data Attribute Store)을 제공하는 컴퓨터 프로그램 코드를 더 포함하는 컴퓨터 판독 가능 매체.

### 명세서

#### 기술분야

본 발명은 일반적으로 컴퓨팅 시스템과 함께 사용하기 위한 데이터 저장 시스템들에 관한 것이며, 특히 비휘발성 메모리 저장 장치 상에 데이터를 유지하는 기술에 관한 것이다.

#### 배경기술

특히, 하나 이상의 집적 회로 칩들에 형성된 플래시 EEPROM(전기적으로 소거 가능하고 프로그래밍 가능한 판독 전용 메모리) 셀의 어레이를 사용하는 소형 팩터 카드 형태의 많은 상업적으로 성공한 비휘발성 메모리 제품들이 오늘날 사용되고 있다. 전형적으로, 개별적인 집적 회로 칩 상의 메모리 제어기가 호스트로의 인터페이스로서 제공된다. 호스트는 제어기를 사용하여 비휘발성 메모리 상에서 다양한 동작들(예를 들어, 판독, 기록)을 수행할 수 있다. 제어기는 전형적으로 마이크로 프로세서, 어떤 비휘발성 판독 전용 메모리(ROM), 비휘발성 랜덤 액세스 메모리(RAM) 및 데이터의 프로그래밍 및 판독 동안 이들이 제어기를 통과할 때 데이터로부터 에러-수정-코드를 계산하는 것과 같은 하나 이상의 특정 회로들을 포함한다. 상업적으로 이용 가능한 카드들 중 일부는 콤팩트플래시™(CF) 카드들, 멀티미디어 카드들(MMC), 보안 디지털(SD) 카드들, 스마트 미디어 카드들, 개인용 태그들(P-태그) 및 메모리 스톱 카드들이다. 호스트들은 개인용 컴퓨터들, 노트북 컴퓨터들, 개인용 디지털 보조장치들(PDAs), 각종 데이터 통신 장치들, 디지털 카메라들, 셀룰러 전화들, 휴대용 오디오 플레이어들, 자동차 음성 시스템들, 및 유사한 유형의 장비를 포함한다. 메모리 카드 구현 이외에, 비휘발성 메모리는 대안적으로 다양한 유형의 호스트 시스템에 내장될 수 있다.

대부분의 집적 회로 애플리케이션들에서와 같이, 어떤 집적 회로 기능을 구현하는데 필요한 실리콘 기판 영역을 줄여야 한다는 압력이 플래시 EEPROM에서도 또한 존재한다. 소정 크기의 메모리 카드 및 다른 유형의 패키지들의 저장 용량을 증가시키거나, 용량을 증가시키고 크기를 감소시키기 위하여 실리콘 기판의 소정 영역에서 저장될 수 있는 디지털 데이터의 양을 증가시키는 것이 계속해서 바람직하다. 데이터의 저장 밀도를 증가시키는 하나의 방법은 메모리 셀당 및/또는 저장 유닛 또는 요소당 1비트 이상의 데이터를 저장하는 것이다. 이는 저장 요소 전하 레벨 전압 범위의 윈도우를 2 이상의 상태들로 분할함으로써 성취된다. 4개의 이와 같은 상태들을 사용하면 각 셀이 2비트의 데이터를 저장하게 되고, 어느 상태든지 저장 요소당 3비트의 데이터를 저장하게 된다. 플로팅 게이트들 및 이들의 동작을 사용하는 다중 상태 플래시 EEPROM 구조는 미국 특허 번호 제5,043,940호 및 제5,172,338호에 서술되어 있고, 유전체 플로팅 게이트들을 사용하는 구조에 대해서는 상술된 미국 특허 출원 공보 제2003/0109093호에 서술되어 있다. 다중-상태 메모리 셀 어레이의 선택된 부분들은 또한 미국 특허 번호 제5,930,167호 및 제6,456,528호에 설명된 방식에서, 다양한 이유들로 인해 2개의 상태들(2진)로 동작될 수 있다.

전형적인 플래시 EEPROM의 메모리 셀들은 함께 소거되는 셀들의 이산 "블록들"로 분할된다. 각각의 소거 블록은 전형적으로 하나 이상의 데이터 페이지들을 저장하는데, 상기 페이지는 하나 이상의 페이지가 상이한 서브-어레이들 또는 플레

인들에서 병렬로 프로그래밍되거나 관독될 수 있을지라도, 프로그래밍 및 관독의 최소 유닛이다. 각각의 페이지는 전형적으로 하나 이상의 데이터 섹터를 저장하는데, 상기 섹터의 크기는 호스트 시스템에 의해 규정된다. 일례의 섹터는 이들이 저장되는 소거 블록 및/또는 사용자 데이터에 관한 어떤 수의 바이트의 오버헤드 정보 이외에, 자기 디스크 드라이브들과 함께 설정되는 표준을 따르는 512 바이트의 사용자 데이터를 포함한다. 이와 같은 메모리들은 전형적으로, 각각의 소거 블록 내에서 32, 128 이상의 페이지들로 구성되며, 각각의 페이지는 하나 또는 단지 몇 개의 호스트 데이터 섹터들을 저장한다. 데이터를 블록에 재기록하기 위하여, 블록이 우선 소거되어야 한다는 점에 또한 유의해야 한다.

메모리 어레이 내로 사용자 데이터를 프로그래밍하고 이 어레이로부터 사용자 데이터를 관독하는 동안 병렬처리 정도를 증가시키기 위하여, 어레이는 전형적으로는, 플레인들이라고 통상적으로 칭하는 서브-어레이들로 분할되는데, 이 서브-어레이들은 자신의 데이터 레지스터들 및 다른 회로들을 포함하여 데이터 섹터들이 동시에 여러 또는 모든 플레인들로 프로그래밍되거나 이 플레인들로부터 관독되도록 병렬로 동작할 수 있다. 단일 집적 회로 상의 어레이가 물리적으로 플레인들로 분할되거나, 각 플레인이 개별적인 하나 이상의 집적 회로 칩들로부터 형성될 수 있다. 이와 같은 메모리 구현의 예들이 미국 특허 제5,798,968호 및 제5,890,192호에 설명되어 있다.

메모리를 더욱 효율적으로 관리하기 위하여, 소거 블록들은 논리적으로 함께 링크되어, 가상 블록들 또는 메타블록들을 형성할 수 있다. 즉, 각 메타블록은 각 플레인으로부터 하나의 소거 블록을 포함하도록 규정된다. 메타블록의 사용은 국제 특허 출원 공개 제WO 02/058074호에 설명되어 있다. 메타블록은 데이터를 프로그래밍하고 관독하기 위한 목적지로서 호스트 논리 블록 어드레스에 의해 규정된다. 마찬가지로, 메타블록의 모든 소거 블록들은 함께 소거된다. 이와 같은 대형 블록들 및/또는 메타블록들과 함께 동작되는 메모리 시스템의 제어기는 호스트로부터 수신되는 논리 블록 어드레스들(LBAs), 및 메모리 셀 어레이 내의 물리적인 블록 번호(PBNs) 간의 변환을 포함하는 다수의 기능들을 수행한다. 블록들 내의 개별적인 페이지들은 전형적으로 블록 어드레스 내의 오프셋들에 의해 식별된다. 어드레스 변환은 종종 논리 블록 번호(LBN) 및 논리 페이지의 중간 항들을 사용하는 것을 포함한다. 따라서, 메모리 어레이의 블록은 단일 소거 블록으로 구성되거나 메타블록을 형성하기 위하여 논리적으로 링크되는 2개 이상의 소거 블록들로 이루어질 수 있다.

호스트 애플리케이션의 관점에서, 플래시 메모리는 논리 어드레스 제로(0)에서 시작하고 논리 어드레스 N으로 끝나는 논리 어드레스 공간으로 분할될 수 있다. 도1은 "논리 섹터들"로 분할되는 호스트의 논리 어드레스 공간(102)을 도시한다. 호스트는 여러 개(예를 들어, 4개에서 64개까지(4-64))의 논리 섹터들을 클러스터로 그룹화한다. 도1에 도시된 바와 같이, 호스트는 논리 어드레스 공간(102)을 클러스터들(A,B,C,D,E,F,G 및 H)을 포함하는 클러스터들로 분할할 수 있다. 이와 같이, 이들 클러스터들 각각은 미리결정된 수의 섹터들(예를 들어, 클러스터당 4개에서 64개(4-64)까지의 논리 섹터들)을 나타낼 수 있다. 어느 경우든지, 호스트는 데이터를 "클러스터"라 칭할 수 있는 논리 데이터 유닛으로 할당한다. 예를 들어, 클러스터들(A, B 및 C)은 파일 1을 할당하는데 사용될 수 있는 반면, 클러스터(D)는 파일 2를 할당하는데 사용된다. 호스트가 클러스터들에서 파일들을 할당한다는 점에 유의해야 한다. 그러나, 클러스터 내의 모든 논리 섹터들이 파일에 대한 데이터를 포함할 필요는 없다. 예로서, 클러스터(C)의 단지 일부(예를 들어, 단지 하나의 논리 섹터)만이 데이터를 가질 수 있다. 그럼에도 불구하고, 호스트는 클러스터 내의 모든 논리 섹터가 필요로 되지는 않을지라도, 전형적으로 전체 클러스터를 파일로 할당한다.

호스트는 파일 할당 테이블(FAT)(104)을 사용하여 파일들을 추적할 수 있다. 파일 할당 테이블(FAT)(104)은 논리 파일 구조를 효율적으로 나타내며, 논리 파일 구조 내의 파일들 각각에 어떤 클러스터가 할당되었는지를 나타낸다. 호스트는 파일 할당 테이블(FAT)(104)을 갱신함으로써 논리 파일 구조를 유지한다. 예로서, 새로운 파일이 할당될 때, 파일 할당 테이블(FAT)(104)은 특히, 새로운 파일을 위해 어떤 클러스터(들)이 할당되었는지를 나타내기 위하여 갱신된다. 또한, 파일 또는 디렉토리가 삭제될 때, 호스트는 파일 할당 테이블(FAT)(104)을 갱신한다. 클러스터들이 갱신의 결과로서 호스트에 의해 할당해제될 수 있다는 점에 유의해야 한다. 예로서, 호스트가 파일 1을 삭제할 때, 갱신된 파일 할당 테이블(FAT)(106)은 호스트가 파일 1을 저장하기 위하여 할당되었던 클러스터들(A, B 및 C)을 효율적으로 할당해제하도록 할 수 있다.

호스트의 논리 어드레스 공간과 대조적으로, 플래시-메모리 제어기의 논리 어드레스 공간은 "논리 블록들"로 분할된다. 다른 말로서, 논리 어드레스 공간(102)은 제어기를 위한 논리 어드레스 공간(103)과 동일하게 표현될 수 있다. 논리 어드레스 공간(102) 및 논리 어드레스 공간(103) 둘 다에 대한 기본적인 유닛은 이 예에서 섹터이다. 도1에 도시된 바와 같이, 논리 어드레스 공간(103)은 논리 블록(1) 및 논리 블록(2)을 포함하는 다양한 논리 블록들로 분할될 수 있고, 각각의 논리 블록들은 메모리 어레이의 단일 블록으로 맵핑되는 다수의 데이터 섹터들을 포함한다. 이들 논리 블록들은 유닛, 예를 들어, 단일 소거 블록 또는 메타블록으로 소거될 수 있는 메모리의 블록들로 맵핑된다. 따라서, 논리 블록들 및 메모리 어레이의 블록들 간에 일대일 대응이 존재하며, 여기서, 논리 블록 내의 모든 섹터들은 단일 블록에 저장된다. 예를 들어, 논리 블록(1)은 블록(2) 내로 맵핑될 수 있고, 이런 방식으로 계속 진행된다. 요약하면, 호스트 및 제어기는 상이한 논리 메모리 유닛을 사용한다. 호스트는 논리 어드레스 공간을 클러스터들로 분할하는 반면, 제어기는 논리 어드레스 공간을 논리 블록들로

분할한다. 전형적으로 호스트 클러스터들이 애플리케이션 데이터 및 서브-디렉토리 엔트리들만을 위해 할당되는 논리적 어드레스 공간에 적용된다는 점에 또한 유의해야 한다. 게다가, 다른 시스템 데이터를 위해 할당된 논리 어드레스 공간은 호스트에 의해 클러스터들로서 관리될 수 없다. 제1 클러스터가 통상적으로 논리 어드레스 공간의 시작부에 할당되지 않는다는 것이 또한 유의되어야 한다. 따라서, 클러스터들 및 논리 블록들은 서로 정렬될 수 없다.

그러나, 종래의 제어기는 호스트에 의해 사용되는 논리 조직을 인지하지 못한다. 통상적으로, 제어기는 호스트에 의해 사용되는 논리 어드레스(102)를 단순히 논리 어드레스(103)에 맵핑하고 나서, 물리적인 어드레스(110)에 맵핑한다. 이는 종래의 제어기가 호스트의 논리 데이터 조직 또는 호스트 동작이 데이터에 대해 가질 수 있는 결과를 고려하지 않는다는 것을 의미한다. 예로서, 새로운 파일이 할당될 때, 논리 파일 시스템(104)은 특히, 새로운 파일을 위해 어떤 클러스터(들)가 할당되었는지를 나타내기 위하여 갱신된다. 또한, 파일 또는 디렉토리가 삭제될 때, 호스트는 논리 파일 시스템(104)을 갱신한다. 더구나, 하나 이상의 클러스터들은 호스트에 의해 할당해제될 수 있다. 예로서, 호스트가 파일 1을 삭제할 때, 갱신된 파일 할당 테이블(FAT)(106)은 호스트가 파일 1을 저장하기 위하여 이전에 할당되었던 클러스터들(A, B 및 C)을 효율적으로 할당해제하였다는 것을 나타낸다(즉, 파일 1은 더 이상 갱신된 파일 할당 테이블(106)에서 표시되지 않는다). 클러스터들(A, B 및 C)이 호스트에 의해 할당해제되고 소거될 수 있지만, 종래의 제어기는 할당해제된 클러스터들(A, B 및 C)에 대응하는 그 데이터를 "가비지 수집(carbage collect)"하지 못한다. 이는 데이터가 메모리에 존속한다는 것을 의미한다. 가비지 수집은 무용 데이터 섹터들(obsolete sectors of data)을 포함하는 원래 블록으로부터의 유효 데이터 섹터들을 갱신 블록으로 복제하는 동작이다. 모든 유효 데이터 섹터가 복사될 때, 원래 블록은 소거를 위해 마킹될 수 있다. 전형적으로, 데이터는 이 방식으로 2개 이상의 블록들로부터 단일 블록으로 통합되므로, 메모리 어레이 내의 공간을 자유롭게 하면서 더 효율적으로 데이터를 저장한다. 가비지 수집 동안의 불필요한 데이터 복제에 의해 이와 같은 동작들의 효율이 감소된다. 게다가, 할당해제된 데이터는 2개의 블록들 내의 데이터가 하나의 블록으로 통합될 수 있을 때, 블록들이 가비지 수집 동안 갱신되는 횟수만큼 복사될 수 있다.

더 설명하기 위하여, 원래 블록(120)이 도1에 도시되어 있다. 간소화를 위해, 원래 블록(120)이 파일들 1 및 2를 할당하기 위하여 호스트에 의해 사용되는 논리 클러스터들(A, B, C 및 D)에 대응하는 데이터를 포함한다라고 가정된다. 또한, 호스트가 나중에 파일 1(클러스터들(A, B, 및 C))을 할당 해제한다라고 가정하면, 원래 블록(120)은 파일 1이 호스트에 의해 할당해제되었다는 것을 제어기가 전혀 인지하지 못하기 때문에, 모든 논리 클러스터들(A, B, C 및 D)을 여전히 포함한다. 그러나, 호스트는 제어기가 원래 블록(120)의 데이터 부분(D)을 갱신할 것을 요청한다. 다른 말로서, 어떤 지점에서, 호스트는 파일 1을 변화시키는 것을 원할 수 있어서, 갱신되어야 하는 하나 이상의 논리 섹터들에 대응하는 데이터에 대해 자신이 인지하는 논리 어드레스를 사용하여 기록 요청을 송신한다. 제어기는 호스트에 의해 어드레스된 이들 논리 섹터들이 호스트(즉, 클러스터(D))에 의해 이미 기록되었던 섹터들에 대응한다라고 결정해서, 갱신 블록(122)이 부분적으로 사용되는데, 그 이유는 원래 블록(120)의 데이터가 겹쳐쓰기 될 수 없기 때문이다. 이는 갱신 블록(122)이 "폐쇄될" 때까지(즉, 원래 블록(120)의 모든 데이터가 블록(122)으로 복제될 때까지), 2개의 블록들이 어떤 시간 동안 논리 블록을 나타내는데 사용될 수 있다는 것을 의미한다. 원래 블록(120) 및 갱신 블록(122)은 예를 들어, 가비지 수집 사이클 동안 결합되어, 갱신 블록(122)이 "원래" 블록(122)이 되고 이전 원래 블록(120)이 소거되어 이용 가능한 메모리 블록들의 풀(pool)로 반환되도록 할 수 있다. 어쨌든, 할당해제된 클러스터들(A, B, 및 C)에 대응하는 데이터가 조만간 갱신 블록(122)으로 복제되어, 갱신 블록(122)이 폐쇄되고 원래 블록(120)이 소거될 수 있다는 점에 유의해야 한다. 제어기는 데이터의 논리 블록들로의 메모리 블록들의 할당을 규정하는 비휘발성 메모리 또는 휘발성 메모리 내에서 맵핑 테이블을 유지할 수 있다. 비-휘발성 메모리에 저장된 데이터를 복제하는 것은 상당한 양의 시간 및 자원이 들게 된다. 그러나, 할당해제된 데이터(예를 들어, 할당해제된 클러스터들(A, B, 및 C))을 복제하는 것은 필요하지 않다. 따라서, 할당해제된 데이터가 복제되지 않는 경우, 시스템 성능은 더욱 향상될 수 있다. 일반적으로, 광범위하게 사용되는 어드레싱 규정을 고수하면서, 시스템 성능을 가능한 한 많이 향상시키는 것이 바람직하다.

따라서, 대안적인 비휘발성 메모리 관리 기술들이 유용할 것이다.

### 발명의 상세한 설명

넓게 말하면, 본 발명은 비휘발성 메모리 시스템(예를 들어, 플래시 메모리)에서 데이터를 관리하는 기술에 관한 것이다. 본 발명의 한 양상에 따르면, 제어기는 호스트의 파일 시스템에 관한 정보를 사용할 수 있는데, 이 정보는 호스트에 의해 비휘발성 메모리 상에 저장되어 하나 이상의 클러스터들(또는 하나 이상의 클러스터들 내의 하나 이상의 섹터들)이 현재 할당되는지를 결정한다. 알 수 있는 바와 같이, 제어기는 특히 호스트의 파일 시스템에 관한 정보를 사용하여 하나 이상의 클러스터들(또는 하나 이상의 섹터들)이 예를 들어 가비지 수집 사이클 동안 메모리 내의 한 위치에서 다른 위치로 복제되지 않도록 한다. 따라서, 종래 수행되는 일부 불필요한 동작들(예를 들어, 데이터 복제)은 피해질 수 있고 시스템 성능은 향상된다.



제어기는 호스트에 의해 파일 할당 테이블(FAT)에 통상적으로 기록되는 정보를 직접 사용하여 하나 이상의 이전 할당된 클러스터가 할당해제되는지를 결정할 수 있다는 것이 또한 인식될 것이다. 대안적으로, 제어기는 루트 디렉토리, 서브디렉토리들 및 파일 할당 테이블(FAT)에 기록된 정보를 사용하여 클러스터 할당에 관한 자신의 정보를 발생하여 예를 들어, 데이터 할당 테이블(DAT)에 유지시킨다. 그러나, 본 발명은 파일 할당 테이블(FAT)에 통상적으로 저장된 정보를 사용하도록 구현될 수 있다는 것이 명백하다. 이와 같이, 제어기가 자신의 정보를 유지시키거나 임의의 부가적인 데이터 구조(예를 들어, 데이터 할당 테이블)를 발생시킬 필요는 없다. 그러나, 후술되는 바와 같이, 데이터 할당 테이블(DAT)을 유지하는 것은 파일 할당 테이블(FAT)로부터 얻어진 정보에만 의존하는 것보다 더 많은 이점을 제공할 수 있다. 그러므로, 예를 들어, 데이터 할당 테이블(DAT)로서 구현될 수 있는 데이터 애트리뷰트(attribute) 스토어(DAS)가 또한 설명될 것이다.

일반적으로, 하나 이상의 데이터 애트리뷰트들을 포함하는 데이터 애트리뷰트 스토어(DAS)는 제어기에 의해 비휘발성 메모리에 유지될 수 있다. 데이터 애트리뷰트는 데이터에 관한 유용한 정보(예를 들어, 할당 또는 할당해제 상태, 크기, 우선 순위)를 제공할 수 있다. 이와 같이, 데이터 애트리뷰트 스토어(DAS)는 데이터를 더 지능적으로 관리하는데 사용될 수 있다. 본 발명의 또 다른 양상은 데이터 애트리뷰트 스토어(DAS)를 사용하여 데이터를 관리하는 기술에 관한 것이다.

하나의 실시예에서, 데이터 애트리뷰트 스토어(DAS)는 할당 스토어(AS)로서 구현된다. 할당 스토어(AS)는 예를 들어, 테이블로서 또는 각 데이터 클러스터(호스트의 논리 유닛)에 할당 상태를 제공하는 데이터 할당 테이블(DAT)로서 구현될 수 있다. 클러스터는 데이터를 국부적으로 조직하기 위하여 호스트가 사용되는 논리 유닛을 표시한다. 더 양호한 메모리 관리를 성취하기 위하여, 비휘발성 메모리 시스템의 제어기는 데이터 할당 테이블(DAT)에 액세스할 수 있다. 이와 같이, 데이터 할당 테이블(DAT)은 제어기 및 호스트에 의해 사용되는 논리 조직 간의 브리지를 효율적으로 제공한다. 이는 제어기가 호스트의 논리 구조를 효율적으로 이해하게 한다. 게다가, 제어기는 호스트 활동도를 감시하여 예를 들어, 호스트가 클러스터를 할당해제하는지를 결정한다. 예로서, 루트 디렉토리, 파일 할당 테이블(FAT) 및 DOS 컴플라이언트 시스템의 서브디렉토리는 감시되어 호스트에 의해 데이터 클러스터의 할당해제를 검출한다. 루트 디렉토리, 파일 할당 테이블(FAT) 또는 서브디렉토리의 변화는 예를 들어, 호스트에 의해 삭제되는 파일의 일부 또는 파일에 따라서 초래될 수 있다. 어쨌든, 하나 이상의 데이터 클러스터들의 할당해제가 검출될 때, 제어기는 데이터 할당 테이블(DAT)을 갱신한다. 따라서, 데이터 할당 테이블(DAT)은 데이터에 관한 갱신된 정보를 제공할 수 있다. 이 정보는 데이터를 더 지능적으로 관리하도록 사용될 수 있다. 예로서, 할당해제 상태는 데이터가 가비지 수집 동안 복제될 수 있는지를 결정하기 위하여 사용될 수 있다. 할당해제된 데이터는 복제될 필요가 없다. 따라서, 통상적으로 수행되는 일부 불필요한 동작들은 피해질 수 있고 시스템 수행 성능은 향상될 수 있다.

다른 실시예에서, 호스트에 의해 FAT 또는 DAS에 저장되는 정보는 호스트에 의해 전송되는 데이터의 일부가 어떻게 저장되는지를 결정하기 위하여 사용될 수 있다. 비순차적으로 수신된 데이터는 비순차적으로 저장될 수 있거나, 대안적으로, 수신된 데이터 내의 임의의 갭들을 채우도록 하는 모든 곳으로부터 데이터를 복제함으로써 순차적으로 저장될 수 있다. 이와 같은 순차적인 저장이 유용한 하나의 경우는 호스트가 논리 어드레스 공간(예를 들어, 클러스터들)의 이용 가능한(자유 또는 할당해제된) 부분들에 기록되는 경우이다. 이와 같은 데이터의 기록은 호스트가 새로운 데이터를 저장한다는 것을 나타내고 비순차적으로 데이터를 저장하기 위한 이유를 전혀 나타내지 않는다. 호스트에 의해 전송되는 데이터 섹터들의 논리 어드레스들을 자유 클러스터들의 논리 어드레스 범위와 비교함으로써, 제어기는 호스트가 다음 자유 클러스터에 기록되는지를 결정할 수 있다. 이 결정을 토대로, 제어기는 섹터들에 대한 저장 방식을 선택할 수 있다.

하나의 실시예에서, 호스트로부터 수신된 섹터 또는 섹터들이 호스트가 다음 자유 클러스터에 기록되는지를 나타내는데에 대한 결정을 행한다. 그렇다면, 데이터는 순차적인 형태로 블록에 기록되는데, 데이터 내의 임의의 갭들은 메모리 어레이 내의 그 밖의 장소로부터 복제되는 데이터에 의해 채워진다. 그렇지 않다면, 순차적으로 또는 비순차적으로 데이터를 저장하는지를 결정하기 위하여 부가적인 기준들이 고려될 수 있다. 예를 들어, 섹터들이 섹터들 간의 논리 어드레스에 점프에 의해 호스트로부터 수신되면, 점프의 크기는 저장 방식을 결정할 수 있다. 점프가 작다면, 이 결과의 갭은 복제된 데이터로 채워질 수 있고, 갱신된 데이터는 순차적인 형태로 유지될 수 있다. 또 다른 예에서, 갱신 블록이 순차적이고 거의 채워진 경우, 갱신 블록은 데이터로 채워져 이를 비순차적으로 수신된 섹터를 기록하는 것이 아니라, 순차적으로 유지시킨다. 저장 체계는 비-휘발성 메모리에서 FAT의 복사로부터 직접 얻어진 정보에 근거하여 선택되거나, FAT로부터 유도된 유사한 구조나 DAS에서의 정보에 근거하여 선택된다.

본 발명은 방법으로서, 시스템, 장치, 기기, 또는 컴퓨터 관독 가능한 매체를 포함하여 여러 가지 방식으로 구현될 수 있다. 본 발명의 다른 특징과 장점은 첨부된 도면을 참조하여, 본 발명의 원리를 예를 들어 예시하여 다음의 상세한 설명으로부터 명백할 것이다.

## 실시예



본 발명은 비-휘발성 메모리 시스템(예를 들어, 플래시 메모리)에서 데이터를 관리하는 기술에 관한 것이다. 본 발명의 특징에 따라, 제어기는 호스트의 파일 시스템에 관련되는 정보를 사용할 수 있고, 이것은 비-휘발성 메모리에서 호스트에 의해 저장되어 하나 이상의 클러스터(cluster: 하나 이상의 클러스터 내에 하나 이상의 섹터)가 현재 할당되는 지를 결정한다. 이해되듯이, 예를 들어 가비지(garbage) 수집 사이클 동안에, 하나 이상의 클러스터(혹은 클러스터 내에 하나 이상의 섹터)가 메모리에서 한 개 위치에서 다른 위치로 복사되지 않도록, 제어기는 다른 것들 사이에 호스트의 파일 시스템에 연관되는 정보를 사용할 수 있다. 그 결과, 통상적으로 실행되는 어떤 불필요한 동작(예를 들어, 데이터 복사)이 회피될 수 있어서 시스템 성능이 향상될 수 있다.

또한, 이전에 할당된 하나 이상의 클러스터가 재할당되었는지의 여부를 결정하도록 상기 제어기가 파일 할당 테이블(FAT)에서 호스트에 의해 정상적으로 기록된 정보를 직접 이용할 수 있다는 것을 이해할 것이다. 대안으로, 상기 제어기는 루트 디렉토리(root directory), 서브디렉토리, 및 파일 할당 테이블(FAT)에 기록된 정보를 이용하여 예를 들어, 데이터 할당 테이블(Data Allocation Table:DAT)에서 클러스터 할당에 대한 그 자체의 정보를 생성 및/또는 유지한다. 그러나, 파일 할당 테이블(FAT)에 정상적으로 저장된 정보를 이용하여 본 발명이 구현된다는 것이 명백해야 한다. 이렇게, 제어기가 그 자체의 정보를 유지하거나 어떤 부가적인 데이터 구조(예를 들어, 데이터 할당 테이블(DAT))를 생성하는 것이 필요하지 않다. 그러나, 아래에 기술되듯이, 데이터 할당 테이블(DAT)의 유지는 파일 할당 테이블(FAT)로부터 얻어진 정보에 오직 의존하기보다는 더 많은 장점을 제공한다. 그러므로, 예를 들어, 데이터 할당 테이블(DAT)로 구현되는 데이터 애트리뷰트 저장(Data Attribute Store:DAS)이 역시 예시된다.

일반적으로, 데이터의 하나 이상의 애트리뷰트를 포함하는 데이터 애트리뷰트 저장(DAS)이 비-휘발성 메모리에서 제어기에 의해 유지될 수 있다. 데이터 애트리뷰트는 데이터에 대한 유용한 정보(예를 들어, 할당 또는 해제 상태, 크기, 우선권)를 제공할 수 있다. 이렇게, 데이터 애트리뷰트 저장(DAS)은 데이터를 더욱 지적으로 관리할 수 있다. 본 발명의 다른 특징은 데이터 애트리뷰트 저장(DAS)을 이용하여 데이터를 관리하는 기술에 관계된다.

일 실시예에서, 데이터 애트리뷰트 저장(DAS)은 할당 저장(Allocation Store:AS)으로서 구현된다. 예를 들어, 할당 저장(AS)은 데이터의 각각의 클러스터(호스트의 논리 유닛)에 대한 할당 상태를 제공하는 테이블 또는 데이터 할당 테이블(DAT)로서 구현될 수 있다. 클러스터는 논리적으로 데이터를 조직하기 위해 호스트가 사용하는 논리 유닛을 나타낸다. 더 양호한 메모리 관리를 얻기 위해, 비-휘발성 메모리 시스템의 제어기는 데이터 할당 테이블(DAT)에 액세스할 수 있다. 이렇게, 데이터 할당 테이블(DAT)은 호스트와 제어기에 의해 사용된 논리 구조 사이에 다리를 효과적으로 제공한다. 이것은 상기 제어기가 호스트의 논리 구조를 효과적으로 이해하도록 한다. 게다가, 제어기는 호스트 활동을 감시하여 예를 들어, 호스트가 클러스터를 해제했는지의 여부를 결정한다. 예로서, DOS 추종 시스템의 루트 디렉토리, 파일 할당 테이블(FAT) 및 서브 디렉토리가 감시되어 호스트에 의한 클러스터의 해제를 검출한다. 루트 디렉토리, 파일 할당 테이블(FAT) 또는 서브 디렉토리의 변화는 예를 들어, 호스트에 의해 제거되고 있는 파일의 부분 또는 파일의 결과로서 일어난다. 어떤 경우에, 데이터의 하나 이상의 클러스터의 해제가 검출되었을 때, 상기 제어기는 데이터 할당 테이블(DAT)을 갱신한다. 이렇게, 데이터 할당 테이블은 데이터에 대한 갱신된 정보를 제공할 수 있다. 이러한 정보는 데이터를 더욱 지적으로 관리하는 데 이용될 수 있다. 예로서, 가비지를 수집하는 동안에 데이터가 복사되어야 하는지의 여부를 결정하도록 해제 상태(deallocation status)가 이용될 수 있다. 해제된 데이터는 복사될 필요가 없다. 그 결과, 통상적으로 실행되는 어떤 불필요한 동작이 회피될 수 있어서 시스템 성능이 향상될 수 있다.

본 발명의 실시예가 도 2A-8을 참조하여 아래 기술된다. 그렇지만, 여기에 주어진 이러한 도면에 대한 상세한 설명이 설명 목적에 있다는 것을 당업자는 쉽게 이해할 것이다. 예를 들어, 기술 내용이 플래시-메모리에 대해 언급되지만 다른 유형의 비-휘발성 메모리도 이용될 수 있다. 다른 유형의 비-휘발성 메모리는 자기-저항 램(magnetoresistive RAM), 페로일렉트릭 램(Ferroelectric RAM) 및 위상 변화 메모리(역시 Ovonic 통일된 메모리 또는 OUM으로 공지)를 포함한다.

위에서 보듯이, 제어기는 호스트의 파일 시스템에 관한 정보를 사용할 수 있고, 이것이 비-휘발성 메모리 위에서 호스트에 의해 저장되어서, 하나 이상의 클러스터(또는 클러스터내 섹터)가 본 발명의 특징에 따라 현재 할당되는지의 여부를 결정한다. 가비지(garbage) 수집 사이클 동안에, 하나 이상의 클러스터(혹은 클러스터 내에 한 개 섹터)가 메모리에서 한 개 위치에서 다른 위치로 복사되지 않도록, 상기 제어기는 다른 것들 사이에 호스트의 파일 시스템에 연관되는 정보를 사용할 수 있다. 예시하기 위해, 도 2A는 본 발명의 특징에 따른 복사 관리(250)를 도시한다. 초기에, 상기 호스트는 파일 시스템에 의한 사용을 위해 정보를 저장(252)하는데 비-휘발성 메모리 장치를 이용한다. 다음에, 하나 이상의 클러스터(또는 하나 이상의 클러스터내의 하나 이상의 섹터)가 현재 어떤 명백한 파일로 할당되는지의 여부를 결정하도록 상기 제어기는 호스트에 의해 저장된(252) 정보를 이용한다. 당업자에 의해 이해되듯이, 하나 이상의 클러스터(또는 하나 이상의 클러스터내의 하나 이상의 섹터)가 현재 어떤 명백한 파일로 할당되는지의 여부를 결정하도록 상기 제어기는 파일 할당 테이블(FAT)에서 호스트에 의해 정상적으로 기록된 정보를 직접 이용한다. 대안으로, 상기 제어기는 장치로 호스트에 의해 기록

된 일련의 논리 섹터를 구비한 루트 디렉토리(root directory), 서브디렉토리, 및 파일 할당 테이블(FAT)에 기록된 정보를 이용하여 클러스터 할당에 대한 그 자체의 정보를 생성 및/또는 유지한다. 예를 들어, 상기 제어기는 그 자체의 데이터 할당 테이블(DAT)을 생성하고/거나 유지한다. 어떤 경우에, 메모리 내에 가비지 수집 동작을 하는 동안에 한 개 위치에서 다른 위치로 복사되지 않는 어떤 클러스터(또는 클러스터 내의 섹터)가 현재 NOT가 할당된다고 결정되는 지를(254) 상기 제어기가 확인한다(256). 환언하면, 한 개 클러스터(또는 클러스터내의 섹터)는 이것이 현재 할당되는지의 여부를(254) 오직 복사한다. 상기 복사 관리 방법(250)은 다음 동작(256)에서 끝난다.

위에서 보듯이, 데이터 할당 테이블(DAT)의 유지는 파일 할당 테이블(FAT)로부터 얻어진 정보에만 의존하기보다는 더 많은 장점을 제공한다. 당업자에 명백해듯이, 상기 FAT는 클러스터가 해제되었다는 것을 항상 가르키지는 않는다. 예를 들어, 클러스터 목록의 조합은 디렉토리에서 어떤 파일 참조점에 연결되지 않고서 FAT에 존재한다.

파일이 삭제될 때, 그의 디렉토리 목록은 파일이 삭제되었다는 것을 나타내려고 호스트(파일이름의 첫째 글자가 0xE5로 변경)에서 파일 시스템에 의해 수정된다. 그러나, 상기 FAT에서 클러스터 목록이 여전히 파일에 할당된 것처럼 그의 연결된 정보를 유지한다. 클러스터를 재사용하고, FAT 클러스터 목록을 사용되지 않는다고 나타내는 "0x0000"로 갱신하는 알고리즘이 호스트에서 특정 파일 시스템 구현의 대표적인 함수이다. 상기 DAT는 디렉토리나 서브 디렉토리 정보에서의 변화를 이용할 수 있어서, 호스트 파일 시스템에 의존하지 않고서 클러스터 애트리뷰트가 클러스터의 현재 할당 상태를 파일로 반영하도록 한다. 어떤 환경에 대해, DAT의 구현이 FAT에 오직 의존하기보다는 더욱 안전하다. FAT가 오직 클러스터의 할당 상태를 확인할 때, 상기 호스트가 파일의 끝에서 FAT를 갱신할 때까지 메모리 시스템이 파일로의 클러스터의 할당에 대해 모른다. 그러므로, 호스트가 이미 기록된 파일의 부분을 수정한다면, FAT에 기록하기 전에, 제어기는 여전히 해제 상태인 어떤 명백한 데이터를 버린다. DAT의 구현은 이러한 이슈를 효과적으로 어드레스 할 수 있다.

게다가, 본 발명의 원리에 따라 제공된 데이터 할당 테이블(DAT)이 FAT에 의해 통상적으로 제공되지 않는 특징을 역시 제공한다. 예를 들어, 상기 FAT는 클러스터에 대한 단일의 할당 상태를 정의한다. 그러나, DAT는 할당 상태가 클러스터 내의 섹터 또는 섹터의 그룹에 풀리도록 허용하여, 예를 들어, 다중-비트 상태가 기록되도록 구현될 수 있다. 이것은 상기 제어기가 파일들 사이에 할당 안 된 섹터의 복사를 회피하도록 하고, 여기서 제 1의 파일의 마지막 클러스터가 오직 부분적으로 이용된다. DAT가 제공하는 장점과 추가적인 특징이 주어져서, 아래에 기술된 실시예가 데이터 애트리뷰트 저장(DAS)에서 그 자체의 정보를 유지하는 제어기를 d예시한다. 그러나, 본 발명의 범위와 정신을 벗어나지 않고서 호스트에 의해 이용된 여러 파일과 연관된 할당 상태를 결정하기 위해 제어기가 FAT에 오직 의존한다는 것은 명백하다.

도 2B는 본 발명의 일 실시예에 따른 계산 환경(200)을 도시한다. 계산 환경(200)에서, 플래시-메모리 제어기(202)는 호스트 어플리케이션(204)이 비-휘발성 메모리(예를 들어, 플래시 메모리 206)에 액세스하도록 한다. 환언하면, 호스트 어플리케이션(204)이 플래시 메모리(206)에 대한 인터페이스로서 플래시 메모리 제어기(202)를 이용한다. 그러기에, 플래시-메모리 제어기(202)가 플래시 메모리(206) 위에서 여러 액세스 동작(예를 들어, 읽기, 쓰기)을 실행하도록 호스트 어플리케이션(204)이 요청할 수 있다. 상기 호스트 어플리케이션(204)이 어드레스 데이터에 논리 어드레스 공간을 이용할 수 있다. 플래시-메모리 제어기는 제어기 논리적 어드레스 공간에 호스트의 논리 어드레스 공간을 매핑하고, 이것이 차례로 플래시-메모리(206) 위에 물리적인 어드레스 공간에 매핑한다.

더욱이, 플래시-메모리는 데이터 애트리뷰트 저장(DAS:208)을 세워서 유지할 수 있다. DAS(208)이 플래시-메모리(206)에 저장된 데이터와 연관된 여러 애트리뷰트에 대한 정보를 포함한다. 예를 통해, 데이터 애트리뷰트 저장(DAS)에 저장된 데이터의 애트리뷰트는 데이터의 할당 상태, 할당된 데이터의 크기, 우선권을 포함할 수 있다. 일반적으로, 데이터 애트리뷰트 저장(DAS: 208)은 어떤 바람직한 데이터 애트리뷰트를 저장할 수 있고, 하나 이상의 비트의 정보로 애트리뷰트를 나타낼 수 있다. 플래시-메모리 제어기(204)는 데이터 애트리뷰트 모니터(208)를 제공하여서 호스트 어플리케이션(204)의 활동을 모니터한다는 것을 알 수 있다. 데이터 애트리뷰트 모니터(208)는 플래시-메모리(206)에 저장된 데이터에 대한 호스트 어플리케이션(204)의 활동을 설명할 수 있다. 이것은 데이터 애트리뷰트가 어디에서 변하여 이에 따라 DAS의 데이터 애트리뷰트를 갱신하는 지를 데이터 애트리뷰트 모니터(208)가 결정할 수 있다는 것을 나타낸다.

보기를 들기 위해, 도 3A는 본 발명의 일 실시예에 따른 데이터 애트리뷰트 저장(DAS: 208)을 도시한다. 본 실시예에서, 데이터 애트리뷰트(208)가 데이터 할당 테이블(DAT:308)로서 구현된다. 이렇게, DAT(308)는 비-휘발성 메모리 시스템의 여러 데이터 부분의 할당/해제 상태에 대한 정보를 제공한다. 알 수 있듯이, 이러한 여러 데이터 부분이 논리 데이터 구조, 즉, 호스트 어플리케이션(204)에 의해 사용된 논리 구조에 따라 조직될 수 있다. 위에서 보듯이, 호스트 어플리케이션이 호스트 논리 유닛으로 데이터를 조직하고, 이것이 클러스터로 언급될 수 있다. DAT(308)는 이러한 클러스터를 나타내는 유닛으로 분할될 수 있다.

이것은 호스트 어플리케이션에 의해 이용된 각각의 논리 클러스터가 DAT(308)에서 목록으로서 나타낼 수 있다는 것을 의미한다. 예를 들어, 클러스터(1)가 제1 목록, 클러스터 10이 제 10 목록 등등으로 나타낼 수 있다 더욱이, DAT(308)의 각각의 목록이 클러스터의 할당/해제를 나타낼 수 있다. 예를 들어, 논리 클러스터가 현재 할당 또는 해제되는지의 여부를 나타내도록 한 개 비트가 DAT(308)의 각각의 목록에 대해 이용될 수 있다. 제1여러 비트가 이용될 수 있어서 여러 애트리뷰트에 대한 정보 및/또는 애트리뷰트에 대한 정보를 제공한다. 어떤 경우에, 상기 DAT(308)는 데이터의 E종은 관리를 허용한다. 예를 통해, 데이터 부분의 할당/해제에 근거하여, 플래시-메모리 제어기는 데이터 복사를 피하고, 이것이 호스트 어플리케이션에 의해 해제되었다.

예시하기 위해, 도 3B는 본 발명의 일 실시예에 따라 DAT(208)의 목록(302)을 도시한다. 테이블 목록(302)이 4비트를 포함하고 제1 비트(비트 0)는 할당/해제 상태를 제공하고 3(3)개의 다른 비트(비트1-3)은 얼마나 많은 데이터가 할당되었는지를 나타낼 수 있다(예를 들어, "100"은 1/2을 나타내고, "111"은 전체를 나타냄). 유사하게, 더 많은 비트가 사용되어 클러스터의 다른 애트리뷰트에 대한 정보를 제공한다. 예를 통해, 각각의 클러스터가 4비트로 나타낼 수 있고, 여기서 1비트는 해제 상태를 나타내고, 3비트는 클러스터의 어떤 부분이 해제되었는지를 나타낸다. 당업자에 명백하듯이, 클러스터에 대한 여러 비트가 여러 다른 방식으로 정의된다. 예를 들어 극성(또는 차수)이 역전되거나 더 많은 비트가 이용될 수 있다(예를 들어, 6, 8, 12). 대안으로, 모든 비트가 할당되어 완전히 해제된 섹터의 "접-클러스터"의 수를 나타내고, 완전히 할당된 클러스터가 예를 들어, 0으로 설정된 모든 비트로 표시될 수 있다.

위에서 보듯이, 데이터 애트리뷰트 모니터(210)는 호스트 어플리케이션(204)의 활동을 모니터한다. 이러한 모니터링에 근거하여, 데이터 애트리뷰트 모니터(259)는 데이터 애트리뷰트 저장(DAS:208)을 갱신할 수 있다. 도 2에 도시된 대로, 호스트 어플리케이션(204)의 활동을 설명하기 위해, 데이터 애트리뷰트 모니터(210)는 데이터의 논리 구조를 구현할 필요가 있고, 이것은 호스트가 이해하고 이용한다. 더 설명하기 위해, 도 3C는 호스트의 논리 데이터 구조를 도시하고, 이것은 본 발명의 실시예에 따라 제어기에 의해 사용된다. 도 3C에 도시되듯이, 호스트는 논리 어드레스 공간을 시스템(310)과 사용자 공간(312)으로 나눈다. 시스템 데이터(310)는 파일 할당 테이블1 (FAT1), FAT2(FAT1의 복사), 및 루트 디렉토리 정보를 포함하고, 이것은 시스템 데이터(310)로서 저장된다. 사용자 공간은 서브 디렉토리에 대한 정보를 포함한다. 데이터 애트리뷰트 모니터(21)는 FAT1, FAT2, 루트 디렉토리, 및 서브 디렉토리를 모니터하여, 호스트 어플리케이션(202)이 어떤 데이터(예를 들어, 파일 제거, 파일을 짧게 함, 등)의 해제 여부를 결정한다.데이터의 모니터링은 역시 도 6을 참고하여 아래 기술된다.

그러나, 도 4에 대해 플래시-메모리 위에 저장된 데이터에 대한 정보를 플래시-메모리 제어기에 제공하는 방법(400)은 본 발명의 일 실시예에 따라 기술된다. 데이터에 대한 정보는 데이터 유지 동안에 데이터에 대한 알려진 결정을 하도록 플래시-메모리 제어기에 의해 이용될 수 있다. 상기 방법(400)이 예를 들어, 도 2의 플래시-메모리 제어기(202)에 의해 이용될 수 있다.

초기에, 플래시-메모리 저장 장치의 제어기에 대해 데이터 애트리뷰트 저장(DAS)이 생성된다(402). 상기 데이터 애트리뷰트 저장(DAS)이 플래시-메모리 저장 장치에서 각각의 다수의 데이터 부분에 대해 하나 이상의 애트리뷰트를 제공한다. 데이터에 액세스하는 데 제어기를 이용하는 호스트의 활동이 모니터 된다(404). 다음에, 모니터링(404)에 근거하여, 플래시 메모리 장치의 데이터 부분에 대한 애트리뷰트가 갱신될지의 여부가 결정된다(406). 이에 따라, 갱신이 플래시-메모리 저장 장치의 적어도 하나의 데이터 부분에 대해 실행되어야 한다고 결정된다면, 적어도 하나의 데이터 부분에 관한 적어도 하나의 애트리뷰트가 갱신된다(408).갱신(408) 뒤에, 호스트의 활동이 모니터링 된다(404). 그리하여, 데이터의 모니터링이 더 이상 필요 없을 때까지 상기 방법(400)이 위에 기술된 유사한 방식으로 진행할 수 있다.

위에 보듯이, 유지 동작을 실행하도록 데이터 애트리뷰트 저장(DAS)이 제공되어 플래시-메모리 제어기에 의해 이용될 수 있다(예를 들어, 가비지 수집). 도 5는 본 발명의 일 실시예에 따라 플래시-메모리 유지 방법(500)을 기술한다. 초기에, 플래시-메모리 제어기가 플래시-메모리 저장 장치 위에 저장된 데이터의 부분 위에 유지동작을 실행할 필요가 있는지의 여부가 결정된다(502). 제어기가 유지 동작을 실행할 필요가 있다고 결정(502)되면, 어떤 부분의 데이터 애트리뷰트 저장(DAS)이 유지되어야 할 데이터의 부분에 대한 정보를 제공하는지가 결정(504)된다. 이에 따라, 데이터 부분에 관한 적어도 하나의 애트리뷰트가 데이터 애트리뷰트 저장(DAS)로부터 판독(506)된다. 마지막으로, 적어도 하나의 데이터 애트리뷰트를 기반으로 동작이 실행(508) 된다. 적어도 하나의 데이터 애트리뷰트에 의해 제공되는 정보에 근거하여, 유지 동작이 더 지적으로 실행될 수 있다는 것을 알아야 한다.

도 6은 본 발명의 일 실시예에 따른 모니터링 방법(600)을 도시한다. 모니터링 방법(600)은 도 4에 예시된 동작을 모니터하고 갱신하는 동안에 실행될 수 있는 어떤 예를 든 작동을 예시한다(즉, 도 4의 동작 404, 406 및 408). 예를 들어, 모니터링 동작(600)은 플래시-메모리 제어기에 의해 실행될 수 있다. 초기에, 기록 동작에 대한 요청이 수신된다고 결정(602)된

다. 다음에, 기입 동작이 표시된 어드레스 공간을 어드레스 할지의 여부가 결정된다. 예를 통해, 표시된 어드레스 공간이 DOS 환경에서 동작하는 호스트에 의해 어드레스 된 논리 어드레스 공간의 서브 디렉토리와 루트 디렉토리, FAT1, FAT2를 포함할 수 있다. 서브 디렉토리에 대해 표시된 어드레스 공간이 루트 디렉토리와 다른 디렉토리에서의 정보로부터 결정된다. 일반적으로, 데이터가 동작(예를 들어, 파일의 디렉토리의 제거 또는 변경, 파일 편집 등)을 해제했다는 것을 호스트 활동이 나타내는 어드레스 공간을 표시된 어드레스 공간이 나타낸다. 어떤 경우에, 기록 동작이 지적된 어드레스 공간에 있다는 동작(604)에서 결정된다면, 지적된 어드레스 공간에 현재 저장된 데이터가 판독(606)된다. 다음에, 기록 동작이 실행(608)된다. 기록 동작이 실행된 후에(608), 이전에 판독(606)된 지난 데이터가 기록된 (608) 데이터와 비교된다. 이에 따라, 데이터에서의 변화가 있는지의 여부가 결정(612)된다. 만약에 데이터로의 변화가 없다고 결정(612) 되면, 기록 동작에 대한 요청이 수신되는지의 여부가 결정(602)된다.

대안으로, 갱신된 데이터가 지난 데이터로부터 다른 위치에 기록되는 플래시 메모리 시스템에서, 이전의 판독(606)을 실행하지 않고서 낡고 새로운 위치에서의 데이터 사이에 비교(610)가 직접 이루어진다.

그러나, 데이터로의 변화가 검출되었다고 결정(612)되면, 그 변화는 설명되어 하나 이상의 클러스터가 해제되었는지의 여부를 결정하도록(616) 해석(interpret)된다(614). 클러스터가 해제되지 않았다고 결정된다면, 기록 동작에 대한 요청이 수신되는지의 여부가 결정(602)된다. 그러나, 하나 이상의 클러스터가 해제되었다고 결정(616)된다면, 데이터 할당 테이블(DAT)에서의 하나 이상의 클러스터의 상태가 “해제된”상태를 나타내도록 변환한다. 따라서, 모니터링 방법(600)은 데이터 할당 테이블(DAT)을 효과적으로 갱신(618)하고, 데이터의 해제를 초래하도록 이때는 표시된 어드레스 공간에서 데이터의 변화가 인지될 때이다.

반면에, 기록 동작이 표시된 어드레스 공간으로 없다고 결정(604)되면, 기록 동작이 실행된다(620). 게다가, 하나 이상의 식별된 클러스터의 상태가 “할당된”으로 설정될 수 있도록, 기록되었던 클러스터가 식별된다(620). 이해되듯이, 이러한 동작이 식별된 클러스터를 나타내는 DAT의 목록이 “할당된”으로 설정되었음을 확인하는 때는 기록 동작이 비-표시된 어드레스 공간에서 실행될 때이다. DAT에 나타난 클러스터의 상태를 “할당된”으로 설정 후에, 기록 동작에 대한 요청이 수신되었는지의 여부가 결정(602)될 수 있다.

위에 보듯이, 예를 들어 데이터 애트리뷰트 저장(DAS)이 유지 동작을 실행하는데 이용된다. 예시하기 위해, 도 7은 본 발명의 일 실시예에 따라 예를 든 가비지 수집 방법(700)을 도시한다. 이해되듯이, 갱신 동작 동안에 블록을 더 잘 갱신하도록 가비지 수집 방법(700)이 데이터 할당 테이블(DAT)을 이용한다. 초기에, 갱신 블록이 닫혀지는 것으로 결정(702)될 때, 갱신 블록의 모든 섹터가 기록되는지의 여부가 결정된다. 갱신 블록의 모든 논리 섹터가 기록되었다면, 갱신 블록은 원래 블록을 효과적으로 대체할 수 있다. 이에 따라, 갱신 블록은 원래 블록으로서 표시(706)되고, 나중에 사용되도록 이전의 원래 블록은 제거(708)된다.

반면에, 갱신 블록의 모든 섹터가 기록되지 않았다고 결정(702)되면, 미 기록된 섹터에 대응하는 논리 섹터가 동일하다(710). 식별된(710) 논리 섹터가 클러스터로 매핑(712)된다. 상기 클러스터가 데이터 할당 테이블(DAT)에서 검사(714)된다. 따라서, 모든 식별된(710) 클러스터가 해제되었는지를 DAT가 나타내는지의 여부가 결정(716)된다. 모든 클러스터가 해제되는 것을 DAT가 나타내면, 원래와 지난 블록이 제거(708)될 때 갱신 블록이 표시된다(706).

모든 클러스터가 해제되었다는 것을 DAT가 나타낼 때(716) 하나 이상의 클러스터가 복사되지 않는다는 것을 알아야 한다. 하나 이상의 클러스터가 여전히 할당되는 것을 DAT 테이블이 나타낼 때만이 갱신 블록에 할당되어 남아 있는 데이터가 복사(718)된다. 어떤 경우에, 갱신 블록이 즉시 또는 나중에 닫힌다는 것을 알아야 한다. 갱신 블록이 표시되어(706) 이전의 원래 블록이 삭제(708)되지만, 할당 안 된 데이터의 불필요한 복사가 피해진다. 가비지 수집 방법(700)이 다음 삭제(708)를 종료한다.

이해되듯이, 플래시-메모리가 본 발명의 일 실시예에 따라 데이터 애트리뷰트 저장(DAS)을 구현하는데 이용될 수 있다. 예시하기 위해, 도 8은 본 발명의 일 실시예에 따라 데이터 할당 테이블(DAT: 800)을 도시한다. 메모리 부분(801)이 데이터 할당 테이블(DAT: 800)을 구현한다. 메모리 부분(801)은 유닛으로서 삭제 가능한 블록이다. 메모리 부분(801)은 데이터 할당 테이블(DAT:800)을 나타내는데 필요한 메모리보다 더 크다. 각각의 섹션은 하나 이상의 페이지에 위치하고, 여기서 한 페이지는 메모리의 프로그래밍의 최소 유닛이다.

도 8에 도시되듯이, 데이터 애트리뷰트 저장(DAS:800)이 메모리 부분(801)의 시작점에 원래 기록되는 복수의 섹션, 즉 원래 섹션 DAS1(802), DAS2(804), DAS3(806) 및 DAS(808)로 분할될 수 있다. 논리적으로 데이터를 조직하는데 호스트에 의해 이용되는 복수의 클러스터를 이러한 원래 섹션의 각각이 나타낼 수 있다. 클러스터와 연관된 데이터 애트리뷰트가 갱신될 필요가 있을 때, 데이터 애트리뷰트 저장(DAS:800)의 해당 섹션이 갱신될 수 있고 이는 데이터 애트리뷰트 저장의

적절한 원래 섹션에 대해 갱신 섹션을 기록함에 의해서이다. 이러한 갱신 섹션은 마지막 원래 섹션 즉DAS4(808) 뒤에 기록된다. 예를 통해서, 새로운 섹션(816)을 기록함으로써 원래 DAS3(806)이 갱신된다. 유사하게, 원래 DAS1(802)를 갱신할 필요가 있을 때, 새로운 섹션(812)이 기록될 수 있다. 원래 DAS4(818) 등을 표시하는데 새로운 섹션(818)이 기록된다.

게다가, 색인 체계가 데이터 할당 테이블(DAT:800)을 유지한다. 도 8에 도시되듯이, 메모리의 각각의 섹션이 색인 부분을 포함한다. 마지막 목록(06)의 이러한 색인 부분이 데이터 애트리뷰트 저장(DAS)의 현재 섹션을 참조한다. 이것은 색인(820, 830, 840 및 850) 각각이 데이터 애트리뷰트 저장(DAS:800)의 제1, 제2, 제3및 제4 섹션을 가르킨다는 것을 의미한다. 현재의 섹션(DAS1, DAS2, DAS3, DAS4)이 원래 섹션으로서 재생되도록 메모리(801)가 반드시 충만할 때, 예를 들어, 메모리 부분(801)은 가비지 수집될 수 있다.

데이터 애트리뷰트 저장(DAS)의 이용이 임의가 되도록 본 발명이 구현될 수 있다는 것을 역시 알아야 한다. 데이터 애트리뷰트 저장(DAS)의 이용이 시스템 기기에 좌우될 수 있고, 특별한 작동 환경을 위해(예를 들어, DOS) 호스트의 실제 논리 구조가 예측된 데이터 구조에 부착되도록 이것이 검증 과정에 의해 결정된다. 대안으로, 데이터 애트리뷰트 저장(DAS)의 이용이 호스트의 명령의 전송에 의해 인에이블되어, 특정의 구동 환경에 대한 데이터 구조와 상응하는 것을 나타낸다. 할당된 클러스터가 해제되자마자 데이터 할당 테이블이 클러스터의 상태(또는 클러스터내의 섹터)를 바꾸는데 필요하다는 것을 알아야 한다. 환언하면, 할당될 때 클러스터가 여전히 할당된다고 믿는 것은 심각한 문제를 일으키지 않는다. 이렇게, 할당 상태를 해제로 설정되게 기다리는 것이 가능하다. 이해되듯이, 다른 시기에 하나씩 보다는 한번에 한 그룹의 클러스터에 대해 할당 상태를 해제로의 설정은 시스템 성능을 개선한다. 반면에, 당업자는 할당된 클러스터에 대한 상태 정보가 항상 엄격히 정확하다는 것을 알 것이다(즉, 할당된 클러스터는 항상 "할당된"으로 표시되어야 한다). 할당된 클러스터가 정확히 표시되도록 하기 위해, 호스트가 클러스터에 데이터를 기록할 때마다 클러스터의 할당 상태가 자동적으로 "할당된"으로 설정될 수 있다. 역시, 최근에 기록된 기록 섹터나 클러스터에, 데이터의 논리 블록에 물리적인 메모리 블록이 할당을 정의하는 매핑 테이블에서의 정보로서 데이터 할당 테이블(DAT)에서의 정보가 동시에 갱신된다는 것을 알 것이다.

게다가, 데이터 애트리뷰트 저장(DAS)이 많은 다른 유지 동작에 대해 이용될 수 있다는 것을 이해할 것이다. 예를 통해, 블록에 해당하는 모든 클러스터가 해제되었다는 것을 데이터 애트리뷰트 저장이 나타낼 때, 블록이 삭제될 수 있다. 예를 들어, 이것은 성능을 개선하는데 배경으로서 작용한다.

다른 실시예에서, 비-휘발성 메모리에서 호스트에 의해 저장된 정보가 호스트로부터 수신된 데이터가 어떻게 저장되는 지를 결정한다. 하나의 실시예에서, FAT와 디렉토리에서 호스트에 의해 저장된 정보가 이용되어 데이터가 순차의 방식 또는 비-순차 방식으로 저장되는지의 여부를 결정한다.

데이터가 순차 또는 비-수차인지의 여부에 좌우하여 호스트로부터 수신된 데이터를 저장하는데 다른 저장 체계를 특정 메모리 시스템이 이용한다. 예를 들어, 미국 특허 출원 번호 10/750,155호 "블럭 관리 시스템을 구비한 비-휘발성 메모리 및 방법"이란 제목의 2003년 12월 30일 출원의 특허 전체가 여기에 참고로 기술되고 이것은 비-순차 방식으로 호스트에 의해 보내지는 데이터를 취급하는 일정한 방법을 기술한다 특히, 순차 갱신 블록에 할당되는 반면에 비-순차 갱신이 데이터가 어떤 순서로 저장되는 갱신 블록에 할당된다. 갱신 블록에서 데이터의 저장에 어떤 유형의 데이터 갱신에 대한 장점을 제공하는 반면에, 다른 유형의 데이터 갱신에 대해 비효율성을 초래한다. 이러한 이유로, 출원 번호 10/750,155호는 소정의 기준이 부합하는 순차적인 방식으로 어떤 비-순차적으로 수신된 데이터를 저장하는 기술을 서술한다. 본 발명의 실시예에서, 데이터가 비-순차적으로 수신될 때 비-휘발성 메모리에 데이터를 순차적으로 또는 비-순차적으로 기록할지의 여부를 결정하는 기준이 비-휘발성 메모리에서 호스트에 의해 저장된 정보의 분석으로부터 결정된다. 2003년 12월 30일에 출원된 "큰 삭제 블록을 구비한 비-휘발성 메모리 시스템의 관리"란 제하의 미국 특허 출원 제 10/749,831호가 여기에 그 전체가 참고로 기재되고, 비-휘발성 메모리 어레이에서 저장을 위해 호스트에 의해 보내진 데이터의 관리의 부가적인 방법을 기술한다. 특히, 비-순차 방식으로 호스트에 의해 갱신된 데이터를 관리하는 방법이 기재된다. 다른 표시된 블록의 소정의 기준에 따른 데이터에 대해 이용된다.

본 발명의 실시예에서, 호스트에 의해 기록되고 있는 데이터 부분(예를 들어, 섹터)의 논리 어드레스가 기록되고 있는 데이터 부분이 다음의 자유(free) 클러스터에서 논리 어드레스를 갖는지의 여부를 결정하도록, FAT, 루트 디렉토리 및 서브 디렉토리(또는 비-휘발성 메모리에 저장된 유사한 호스트 데이터)와 비교되거나, DAS에 저장된 정보 같은 유도된 정보와 비교된다. 데이터 부분이 다음의 자유 클러스터에서 어드레스를 갖는다면, 이것은 호스트가 다음의 가용한 위치에서 새로운 데이터를 저장한다는 것을 나타낸다. 이같은 저장 패턴으로, 데이터를 순차적으로 저장하고, 비-휘발성 메모리에 이미 저장된 데이터를 복사하므로써 갱신 블록에 어떤 갭을 채우는 것이 바람직하고, 비-순차적인 순서로 데이터를 저장하는데 대해 반대로, 순차적으로 저장되는 장소에 그것을 나중에 복사한다.

도 9A는 FAT 테이블(900)의 예를 도시한다. 각각의 클러스터 0-8은 파일에서 다음의 클러스터를 나타내는 FAT테이블에서의 목록을 구비한다. 이렇게, 클러스터 0에 대한 목록이 "1"로서 이러한 파일에 대한 다음 클러스터가 클러스터 1에 있다는 것을 나타낸다. 클러스터 1에서의 목록은 "5"로서 다음 클러스터가 클러스터 5라는 것을 나타낸다. 클러스터 5에서의 목록은 "6"으로 다음의 클러스터가 6이라는 것을 나타낸다. 클러스터 6에서의 목록이 "EOF"로서 "파일의 끝"을 나타내거나 클러스터가 이러한 파일에 대해 데이터를 포함하지 않는다는 것을 나타낸다. 다른 파일이 유사하게 클러스터 3과 4에 저장된다. 클러스터 4, 7과 8은 "자유(free)"로 혹은 할당 안 된 것으로 나타난다. 이것은 예를 들어, 파일이 안 쓰이는 호스트에 의한 해제의 결과이다. 데이터가 비-휘발성 메모리에서 소거되는 것을 호스트에 의한 해제가 반드시 의미하지 않는다는 것을 알아야 하고, 파일을 저장하는데 사용된 논리 어드레스 범위가 새로운 데이터의 저장을 위해 가용하게 된다. 안 쓰는 데이터는 나중의 시간까지 비-휘발성 메모리에 저장되어 물리적으로 남게 된다. 안 쓰는 제어 정보를 포함하는 클러스터가 역시 해제된다. 비-휘발성 메모리에서 호스트에 의해 역시 저장된 루트 디렉토리 또는 서브 디렉토리가 파일의 제1 클러스터를 나타낸다. 도 9B는 루트 디렉토리와 서브 디렉토리에 포함된 정보와 함께, FAT(900)로부터 유도된 DAT(910)를 도시한다. 클러스터가 할당되거나 자유(free)인지의 여부를 나타내는 각각의 클러스터에 대한 목록을 DAT(910)가 포함한다. 이전에 기술된 대로 부가적인 데이터가 역시 DAT(910)에 저장된다. DAT에 저장된 정보가 FAT와 디렉토리에서의 정보보다 더 현실적이 되도록 DAT가 유지된다. 갱신된 FAT 또는 디렉토리 정보가 호스트에 의해 기록되기 전에, 호스트가 저장을 위해 새로운 데이터를 보내자마자 DAT가 갱신된다. 유사하게, DAS의 다른 정보가 통용되도록 유지된다.

도 9C는 FAT(900)의 논리 어드레스 범위의 부분에 대해 섹터와 클러스터 사이의 통신을 도시한다. 본 예에서, 클러스터는 4 섹터의 데이터를 갖는다. 다른 개수의 섹터를 갖는 클러스터가 역시 이용된다. 통상적으로, 클러스터는 4와 64 섹터 사이에 포함된다. 클러스터 4는 섹터 X, 섹터 X+1, 섹터 X+2 및 섹터 X+3을 구비한다. 호스트에 의한 논리 어드레스의 할당이 일반적으로 매 클러스터를 기반으로 이루어지지만, 메모리 제어기는 일반적으로 섹터에서의 데이터를 취급한다. 일반적으로, 클러스터가 단일 유닛으로 할당되어, 클러스터의 전체 논리적 어드레스 범위가 데이터의 저장에 대해 이용되지 않을지라도 클러스터의 전체 논리 어드레스 범위가 다른 데이터의 나중의 저장에 대해 가용하지 않게 된다. 도 9C는 음영이 진데로 할당된 클러스터 5와 6으로부터의 섹터를 도시하여 이러한 논리 어드레스가 데이터의 저장에 대해 호스트에 가용하지 않다는 것을 나타낸다.

도 9D와 9E는 비-순차적으로 호스트로부터 수신된 데이터의 저장을 위한 두 개의 교대의 체계를 도시한다. 양 쪽 도면은 비-휘발성 메모리에서 저장을 위해 호스트로부터 수신되고 있는 섹터 X내지 X+3과 X+12내지 X+15를 도시한다. 이러한 섹터는 도 9C의 클러스터 4와 7에 해당한다. 도 9D에서 수신된 데이터는 갱신 블록(920)에 기록된다. 섹터 X+12가 섹터 X+3뒤에 바로 기록되기에 이러한 블록은 케이오틱(chaotic)하여 섹터 X+3에서 X+12까지 논리 어드레스 범위의 점프가 있다. 케이오틱(chaotic) 블록(920)은 그것이 섹터 X내지 X+3을 포함할 때 순차적 갱신 블록으로 시작하여 섹터 X+12가 저장될 때 케이오틱(chaotic) 갱신 블록으로 변환된다. 일반적으로, 호스트가 비-순차적인 방식으로 데이터를 보내는 경우에, 케이오틱(비-순차적) 블록이 이용된다. 도 9D의 케이오틱 블록(920)에서의 명백한 데이터가 마침내 순차적인 블록(922)으로 재할당되고, 여기서 그것은 섹터 X+4내지 X+11로 저장되고, 이것이 다른 위치로부터 복사된다. 이것은 일반적으로 통합 동작의 일부로서 취해진다. 순차적인 블록은 일반적으로 장기 저장에 대해 바람직하고, 이는 섹터가 소정의 방식으로 배열되어 블록내에 섹터의 색인을 요하지 않기 때문이다. 케이오틱 블록(920)의 모든 명백한 데이터가 다른 위치로 복사된 뒤에, 케이오틱 블록(920)이 안 쓴 것으로 표시되어 소거된다.

도 9E는 비-순차 데이터, 순차 블록(930)에서 순차적인 방식으로 갱신되는 섹터 X내지 X+3과 X+12내지 X+15를 도시한다. 블록(930)이 이지점에서 순차블록이도록 섹터 X내지 X+3이 먼저 기록된다. 섹터 X+12가 수신될 때, 그것이 비록 비-순차적으로 수신되더라도 데이터를 순차적으로 저장하도록 결정이 이루어진다. 호스트로부터 수신된 섹터사이에 논리 어드레스 범위를 채우도록 원래 블록으로부터 섹터 X+4내지 X+11(클러스터 5와 6에 해당)을 복사하여 순차적인 저장에 얻어진다. 섹터 X+4내지 X+11은 속을 채워 넣는 섹터로 고려되고 이는 그것들이 호스트에 의해 보내진 섹터에서 갭(gap)을 채우거나 "늘리기" 때문이다. 섹터 X+4 내지 X+11은 다른 위치에서 복사된 데이터의 명백한 섹터이다. 이런 식으로, 수신된 섹터가 케이오틱 블록에서 중간 저장 단계를 통해 나가지 않고 순차적인 블록(930)에서 저장된다. 이렇게, 상기 두 개의 기술 사이의 주요한 차이는 섹터 X내지 X+3과 X+12내지 X+15의 기록과 나중에 케이오틱 블록(920)의 소거가 도 9E의 예에서 불필요하다. 섹터 X내지 X+3과 X+12내지 X+15는 한번 기록되고, 섹터 X+4내지 X+12는 이 예에서 여전히 복사된다. 이것은 8개의 더 작은 섹터 기록과 한번의 더 작은 소거 동작을 요하여, 데이터 저장의 효력을 개선한다.

비-순차적으로 수신된 데이터를 순차적으로 혹은 비-순차적으로 저장하느냐의 결정은 여러 기준에 근거한다. 비-순차적인 저장에 일정한 데이터가 더 잘 맞고, 이내 다시 갱신되어야 할 것 같은 데이터가 특히 더 잘 맞는다. 예를 들어, 제어 데이터나 호스트 데이터의 자주 갱신되는 부분이 케이오틱 블록에 더 효과적으로 저장된다. 일정한 데이터가 순차적인 저장에 더 잘 맞고, 데이터가 이내 갱신되어야 할 것 같지 않다면 특히 더 잘 맞는다. 비-순차적으로 수신된 데이터가 일반적으로



비-순차적으로 저장되고, 어떤 경우에 그것을 순차적으로 기록하는 것이 바람직하다. 케이오틱 블록에서 중간의 저장에 회피되기에, 순차적인 형식으로 데이터를 얻는데 케이오틱 블록을 가비지 수집할 필요가 없다. 이것은 도 9D에 도시된 통합 동작을 피하여서 오버헤드(overhead)를 줄이고 성능을 개선한다. 호스트가 다음의 자유 클러스터에 기록할 때, 호스트는 다음의 가용한 위치에서 새로운 데이터를 간단히 기록한 것을 표시하고 논리 어드레스 범위가 이내 다시 갱신되리라는 것을 일반적으로 표시하지 않는다. 그러므로, 도 9E에 도시된 것 같은 순차적인 갱신 체계의 선택은 도 9D에 도시된 것과 같은 케이오틱 갱신 체계의 선택보다는 더 효과적이다. 이렇게 데이터가 다음의 자유 클러스터 출신인지의 여부에 따라 저장 체계가 결정된다. 데이터가 도 9E에 도시된 다음의 자유 클러스터 출신일 때, 순차적인 저장이 선택된다. 이렇게, 도 9D의 저장 체계가 본 예에서 이러한 데이터에 대해 선택되지 않는다.

도 10은 본 발명의 실시예에 따른 저장 체계의 선택을 도시한 흐름도이다. 비-순차 데이터가 호스트로부터 수신될 때, 데이터가 다음의 자유 클러스터(10) 출신인지 여부의 결정이 제어기에 의해 이루어진다. 이러한 결정은 비-휘발성 메모리에서 호스트에 의해 저장된 디렉토리 정보와 FAT를 직접 보거나 DAS나 DAT에 저장된 정보같은 디렉토리 정보와 FAT로부터 유도된 정보를 봄으로써 이루어진다. 현재 어드레스된 클러스터가 다음의 자유 클러스터가 아니라면, 디폴트 체계(12)에 따라서 데이터가 정상적인 방법으로 저장된다. 이것은 데이터가 비-순차적으로 저장되는 것을 의미하거나, 다른 예에서 순차적으로 기록되도록 하는 다른 기준이 있다. 현재 어드레스 된 클러스터가 다음의 자유(free) 클러스터라면, 순차적인 기록이 선택된다. 이런 경우에, 마지막으로 기록된 클러스터와 현재 어드레스 된 클러스터 사이의 논리 어드레스 갭을 메꾸려고 데이터의 명백한 클러스터가 원래 메타블럭(metablock)에서 현재의 순차적인 메타블럭(14)으로 복사된다. 다음에, 현재 어드레스 된 클러스터로부터 수신된 데이터가 현재의 순차적인 메타블럭으로 기록된다. 이렇게, 현재의 메타블럭(16)이 순차적인 것으로 남는다.

도 11은 일 실시예에서 호스트로부터 수신된 섹터에 대한 저장 체계의 선택을 보여주는 흐름도이다. 유사한 저장 체계 선택 흐름도가 출원 번호 10/750,155호에 도시된다. 열린 갱신 블록(20)이 있다면, 갱신 블록이 순차적(22)이고 갱신 블록(24)에서 섹터가 마지막 섹터까지 순차적이고, 다음에 섹터가 갱신 블록까지 기록(26)된다. 갱신 블록에서 섹터가 마지막 섹터까지 순차적이지 아니라면, 강제된 순차 기록이 실행(27)되어야 하는지의 여부에 대한 결정이 이루어진다.

갱신된 블록 내의 최종 섹터로부터 수신된 섹터로의 어드레스 점프가 소정 량( $C_B$ )(28)보다 크지 않다면, 최종 섹터 및 수신된 섹터 간의 갭은 복제된 섹터들(30)로 패딩되어 갱신된 블록을 상술된 바와 같이 순차적인 블록으로 유지시킨다. 갱신 블록에 채워지지 않은 물리적인 섹터들의 수가 소정량( $C_C$ )(32)보다 크지 않다면, 갱신 블록은 폐쇄되어(34) 이를 순차적인 블록으로서 유지하고 새로운 갱신 블록은 상술된 바와 같이 할당될 수 있다(36). 강제 순차 기록을 야기하는 제3의 경우는 호스트가 다음 이용 가능한 클러스터(40)에 기록될 때이다. 이와 같은 기록이 갱신 블록 내의 수신된 섹터 및 최종 섹터 간의 갭에서 발생될 때 비휘발성 메모리(30)의 그밖의 모든 곳으로부터 복제되는 섹터들로 패딩되어, 갱신 블록이 순차 블록으로서 유지될 수 있도록 한다. 순차적인 형태로 데이터를 기록하기 위한 이들 기준들 중 어느 것도 부합되지 않으면, 갱신 블록은 케이오틱 갱신 블록(42)으로 변환될 수 있다. 따라서, 저장 방식을 선택하기 위한 선택 기준들은 호스트에 의해 비휘발성 메모리에 저장된 데이터로부터 도출되는 애플리케이션을 포함할 수 있다.

도 12A는 갱신 블록(50)에 순차적으로 데이터를 저장하는 메모리 시스템에 의해 호스트로부터 비순차적으로 수신되는 데이터의 예를 도시한 것이다. 클러스터(2)의 섹터들(A)은 수신되고 클러스터(7)의 섹터들(B)은 순차적으로 수신된다. 섹터들(A)이 수신될 때, 이들은 갱신 블록(50)의 제1 섹터일 수 있고 갱신 블록(50)은 섹터들만을 포함하는 경우 순차적이라고 간주된다. 대안적으로 도 12A에 도시된 바와 같이, 섹터들(A)이 갱신 블록(50)에 사전 저장된 섹터들과 함께 순차적으로 저장되기 때문에 갱신 블록(50)은 순차적일 수 있다. 갱신 블록(50)이 섹터들(B)을 수신하기 전에 순차적이라고 가정하면, 섹터들(B)이 갱신 블록(50)을 순차적인 갱신 블록으로서 유지하거나 갱신 블록(50)(섹터들 A)에 기록되는 최종 섹터들 직후 섹터들(B)을 저장하는지에 대해 섹터들(B)이 수신될 때를 결정함으로써, 갱신 블록(50)을 케이오틱 블록으로 변환시킨다. 이 예에서, 클러스터(7)는 섹터들에 이은 다음 자유 클러스터이다. 그러므로, 클러스터(7)로부터 섹터들(B)이 수신될 때, 강제 순차적인 기록이 발생되고 섹터들은 갱신 블록(50)에 복제되어 섹터들(A) 및 섹터들(B) 간의 갭을 채운다. 따라서, 도 12A의 갱신 블록(50)에서, 데이터는 순차적으로 저장된다.

도 12B는 갱신 블록(52)에 데이터를 비순차적으로 저장하는 메모리 시스템에 의해 호스트로부터 비순차적으로 수신되는 데이터의 예를 도시한다. 섹터들(A)이 도 12A에서처럼 우선 수신된다. 섹터들(A)은 순차적으로 저장되어, 섹터들(A)이 프로그램될 때 갱신 블록(52)이 순차적이 되도록 한다. 그 후, 섹터들(X)은 수신된다. 섹터들(X)은 다음 자유 클러스터에 기록되지 않는다. 섹터들(X)은 할당된 클러스터에 기록됨으로 강제된 순차 기록을 수행하지 않는다. 섹터들(X)은 비순차적으로 기록될 수 있거나 이와 같이 행할 어떤 다른 이유가 있다면(예를 들어, 섹터들(X) 및 섹터들(S) 간의 논리 갭이 임계값보다 낮다면) 순차적으로 기록될 수 있다. 여기서, 섹터들(X)은 갱신 블록(52)에서 다음 이용 가능한 공간으로 기록되는 것으로 도시된다. 따라서, 갱신 블록(52)은 케이오틱이 된다. 도 12C의 제2 예에서, 섹터들(Y)은 섹터들(A) 다음에 수신된다.

갱신 블록(54)은 섹터들(Y)이 수신되기 전, 순차적이 된다. 섹터들(Y)은 다음 자유 클러스터로부터 나오지 않는데, 그 이유는 갱신 블록(섹터들 A)에 최종 기록된 섹터 및 섹터들(Y) 간에 자유 클러스터(클러스터 7)가 존재하기 때문이다. 그러므로, 섹터들(Y)은 강제 순차 기록을 야기하지 않고 이들은 패딩 없이 갱신 블록(54)에 기록되어, 갱신 블록(54)이 케이오틱이 되도록 한다. 도12D에 도시된 제3 예에서, 섹터들(Z)은 섹터들(A) 다음에 수신된다. 갱신 블록(56)은 섹터들(Z)의 수신 전, 순차적이 된다. 섹터들(Z)은 클러스터 0으로부터 나와, 이들은 그 이유는 클러스터 0이 클러스터 2 전에 있기 때문에 다음 자유 클러스터로부터 나오지 않음으로, 도시된 순서가 제공되면, 클러스터 0은 다음 자유 클러스터가 되지 않는다. 따라서, 섹터들(Z)은 클러스터들(A) 직후에 기록됨으로써, 갱신 블록(56)이 케이오틱이 되도록 한다.

본 발명의 장점은 수없이 많다. 상이한 실시예들 또는 구현방식들이 다음 장점들 중 하나 이상을 발생시킬 수 있다. 이것의 전부가 아니고 본원에 서술되지 않은 다른 장점들이 있을 수 있다. 본 발명의 한 가지 장점은 비휘발성 메모리 시스템의 수행능을 개선한다는 것이다. 본 발명의 또 다른 장점은 폭넓게 사용되는 협약을 사용하여 구현될 수 있다는 것이다. 본 발명의 또 다른 장점은 플래시 메모리 내의 상대적으로 작은 애트리뷰트 테이블로서 구현될 수 있다는 것이다. 다른 구현방식들에서, 테이블이 필요로 되지 않는다. 또 다른 장점은 본 발명이 배경 유지 동작 동안 사용하기 위하여 및/또는 임의의 사용을 위하여 구현될 수 있다는 것이다.

상술된 본 발명의 각종 양상들 또는 특징들은 단독으로 다양한 조합으로 사용될 수 있다. 본 발명은 또한 하드웨어 또는 하드웨어 및 소프트웨어의 조합에 의해 구현될 수 있다. 본 발명은 역시 컴퓨터 판독 가능한 매체 상에 컴퓨터 판독 가능 코드로 구현될 수 있다. 컴퓨터 판독 가능한 매체는 데이터를 저장할 수 있는 어떤 데이터 저장 장치로서, 이것은 컴퓨터 시스템에 의해 판독될 수 있다. 컴퓨터 판독 가능한 매체의 예는 판독 전용 메모리, 랜덤-액세스 메모리, CD-롬, DVD, 마그네틱 테이프, 광학 데이터 저장장치 및 캐리어 웨이브를 포함한다. 역시 컴퓨터 판독 가능한 매체는 네트워크가 연결된 컴퓨터 시스템에 분산되어 분산된 형식으로 컴퓨터 판독 가능한 코드가 저장되어 실행될 수 있다.

본 발명의 많은 특징과 장점이 상세한 설명에서 명백하고, 이렇게 첨부된 특허 청구범위에 의해 본 발명의 그러한 특징과 장점이 지원된다. 더욱이, 여러 가지의 변경과 수정이 당업자에게 일어날 수 있기에, 본 발명을 도시되고 기술된대로 정확하게 구조와 동작을 제한하는 것은 바람직하지 않다. 그리하여, 모든 적당한 수정과 등가물이 본 발명의 범위 내에서 이루어질 수 있다.

### 도면의 간단한 설명

도 1은 여러 논리 섹터로 분할될 수 있는 논리 어드레스 공간을 예시한다.

도 2A는 본 발명의 일 실시예에 따른 복사 관리 방법을 도시한다.

도 2B는 본 발명의 일 실시예에 따른 컴퓨팅 환경을 도시한다.

도 3A는 본 발명의 일 실시예에 따른 데이터 할당 테이블(DAT)을 도시한다.

도 3B는 본 발명의 일 실시예에 따른 데이터 할당 테이블(DAT)의 엔트리를 도시한다.

도 3C는 본 발명의 일 실시예에 따른 제어기에 의해 사용되는 호스트의 논리 데이터 구조를 도시한다.

도 4는 본 발명의 일 실시예에 따른 플래시-메모리에 저장된 데이터에 대한 정보를 갖는 플래시-메모리 제어기를 제공하는 방법을 도시한다.

도 5는 본 발명의 일 실시예에 따른 플래시-메모리 유지 방법을 예시한다.

도 6은 본 발명의 일 실시예에 따른 모니터링 방법을 예시한다.

도 7은 본 발명의 일 실시예에 따른 예를 든 가비지(garbage) 수집 방법을 도시한다.

도 8은 본 발명의 일 실시예에 따른 데이터 할당 테이블(DAT)을 도시한다.

도 9A는 파일 할당 테이블(File Allocation Table)을 도시한다.

도 9B는 도 9A의 FAT로부터 유도된 DAT를 도시한다.

도 9C는 호스트 데이터의 부분에 대한 클러스터와 섹터 사이의 통신의 예를 도시한다.

도 9D는 캐시된 블록에 저장되고 있으며 나중에 순차 블록에 복사되는 비-순차 데이터를 도시한다.

도 9E는 순차 블록에서 직접 저장되고 있는 비-순차 데이터를 도시한다.

도 10은 본 발명의 실시예에 따른 호스트 데이터의 부분에 대한 저장 체계를 선택하는 흐름도를 도시한다.

도 11은 본 발명의 다른 실시예에 따른 호스트 데이터의 부분에 대한 저장 체계를 선택하는 흐름도를 도시한다.

도 12A는 순차적으로 저장되는 비-순차적인 데이터의 예를 도시한다.

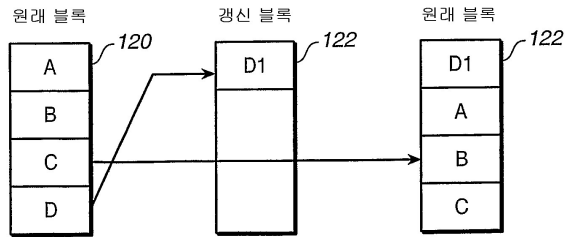
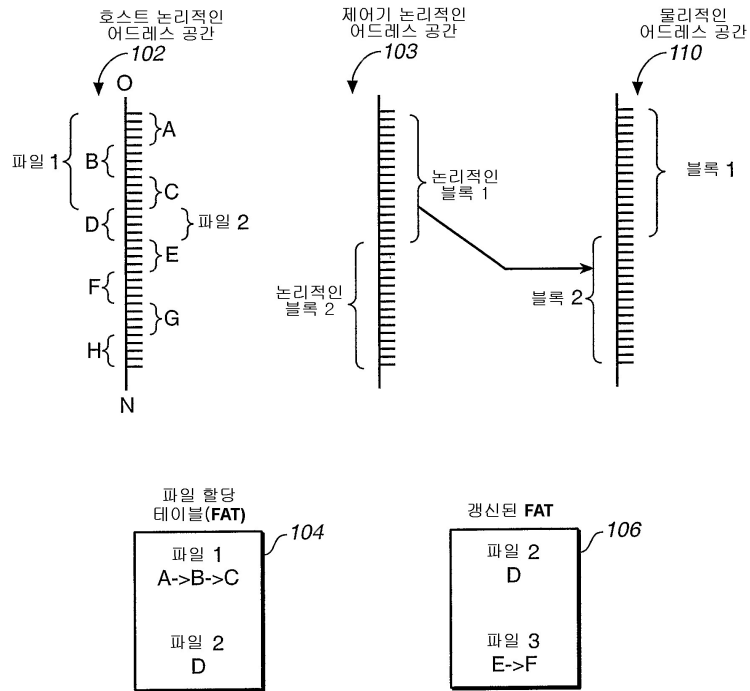
도 12B는 비-순차적으로 저장된 비-순차적인 데이터의 제1 예를 도시한다.

도 12C는 비-순차적으로 저장된 비-순차적인 데이터의 제2 예를 도시한다.

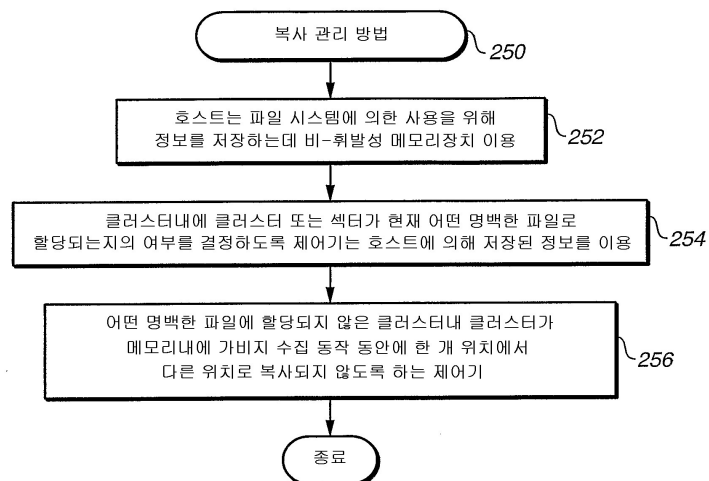
도 12D는 비-순차적으로 저장된 비-순차적인 데이터의 제3 예를 도시한다.

도면

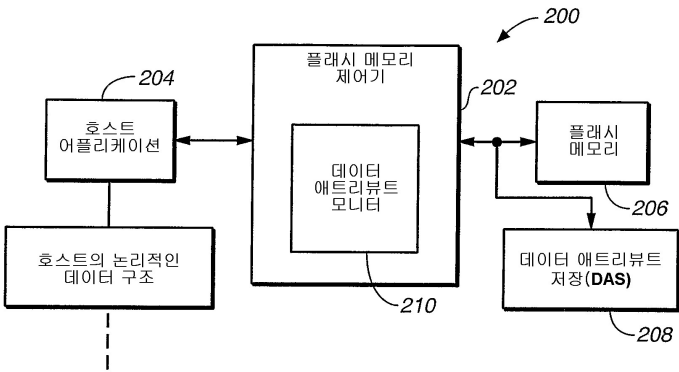
도면1



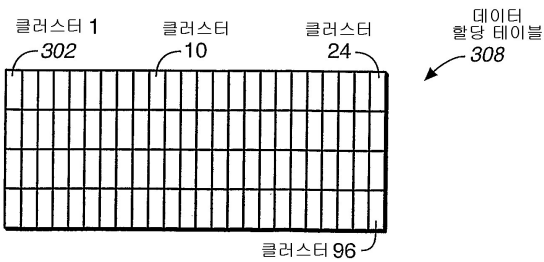
도면2a



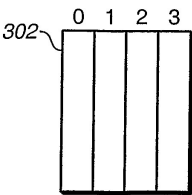
도면2b



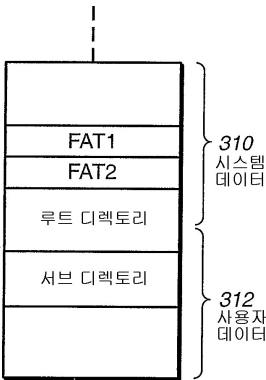
도면3a



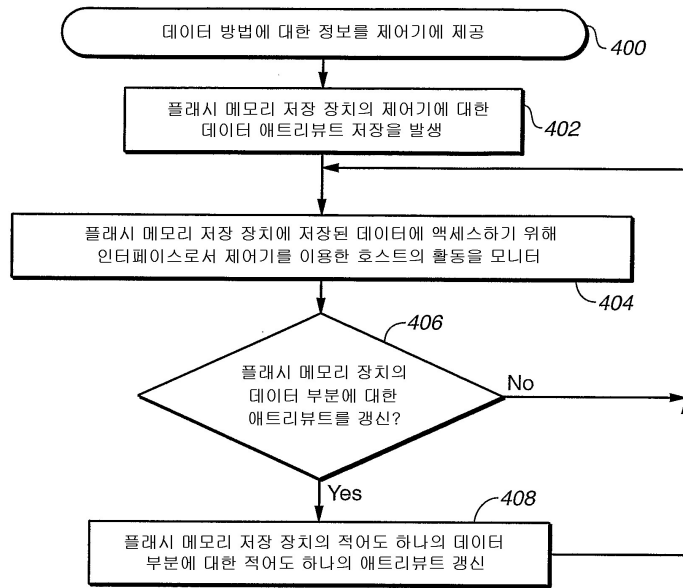
도면3b



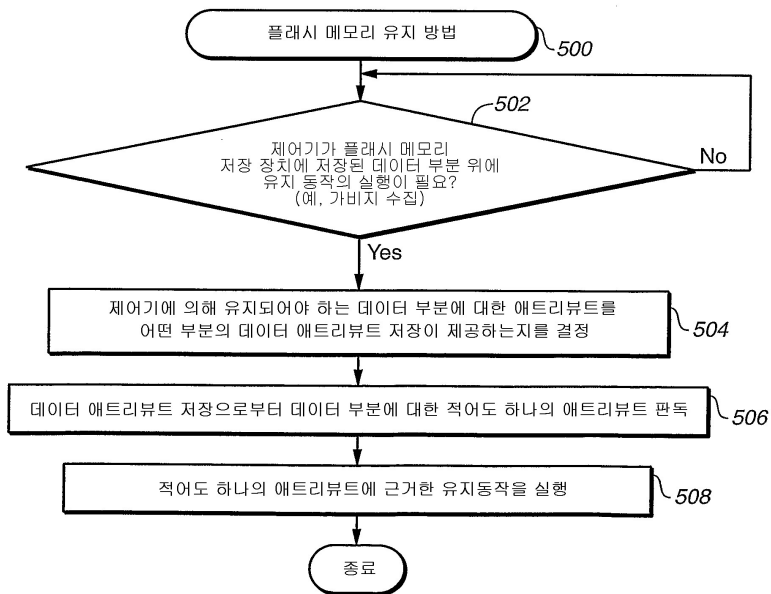
도면3c



도면4

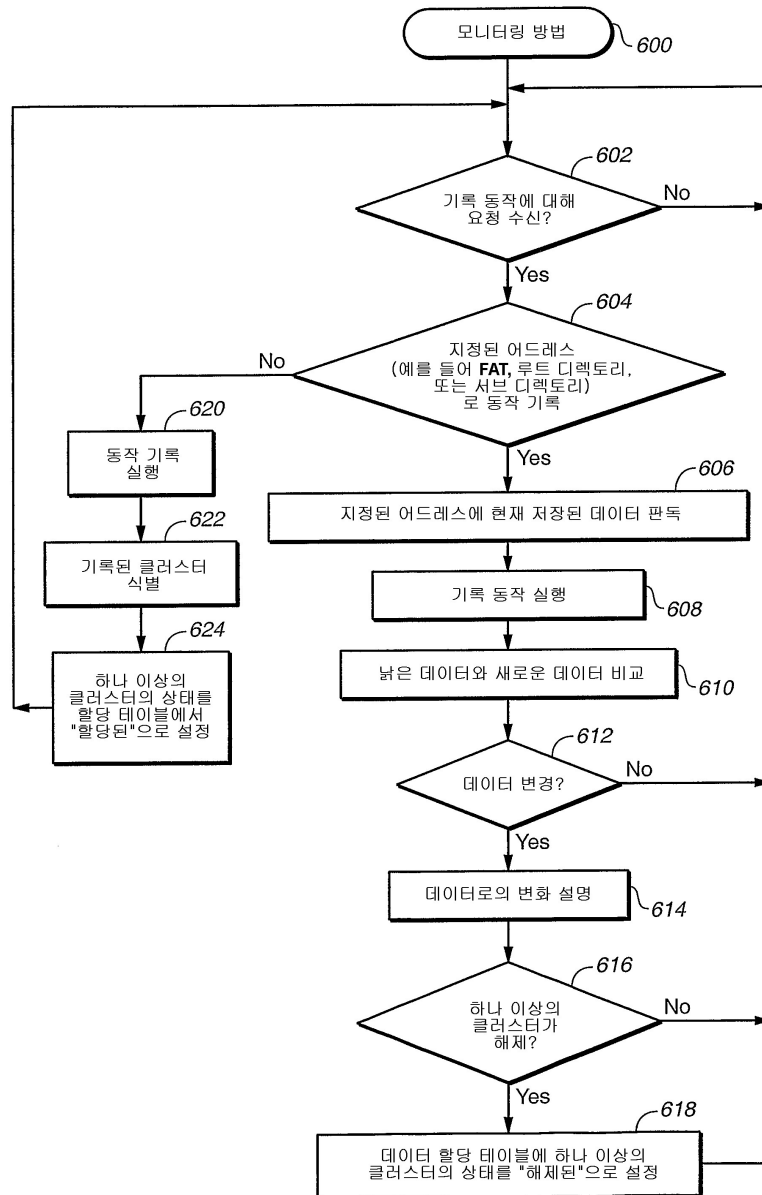


도면5

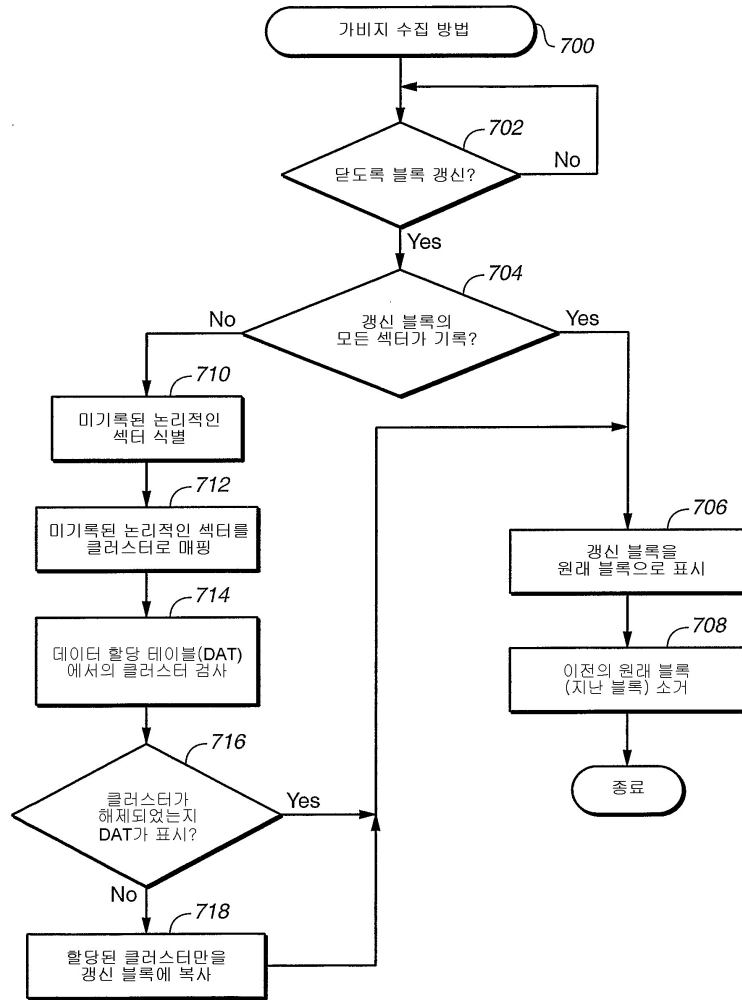




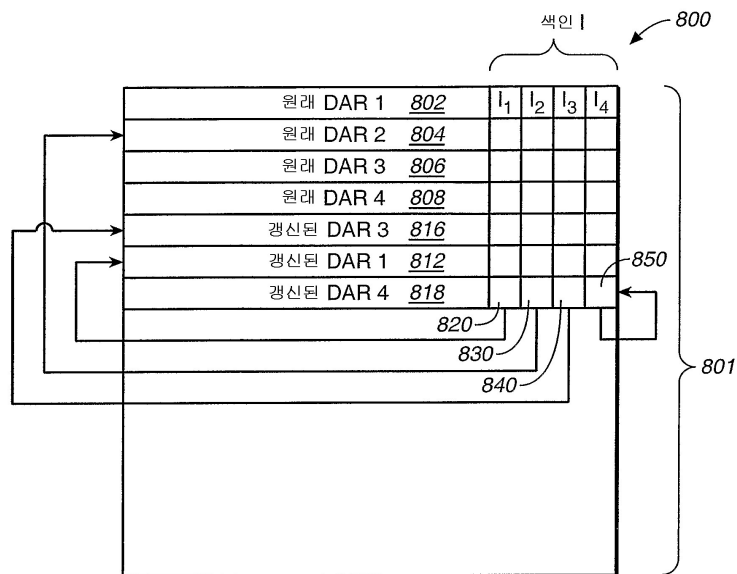
도면6



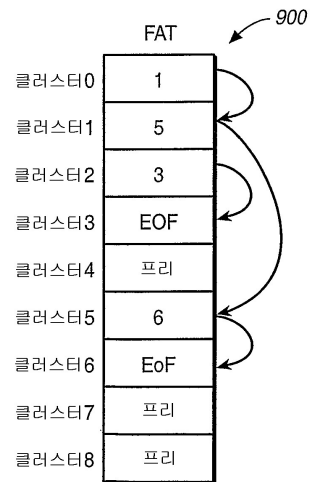
도면7



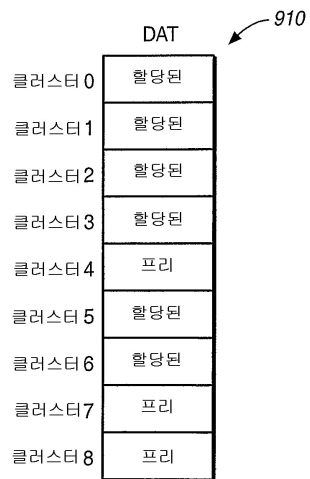
도면8



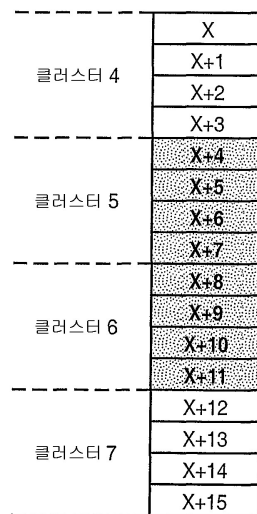
도면9a



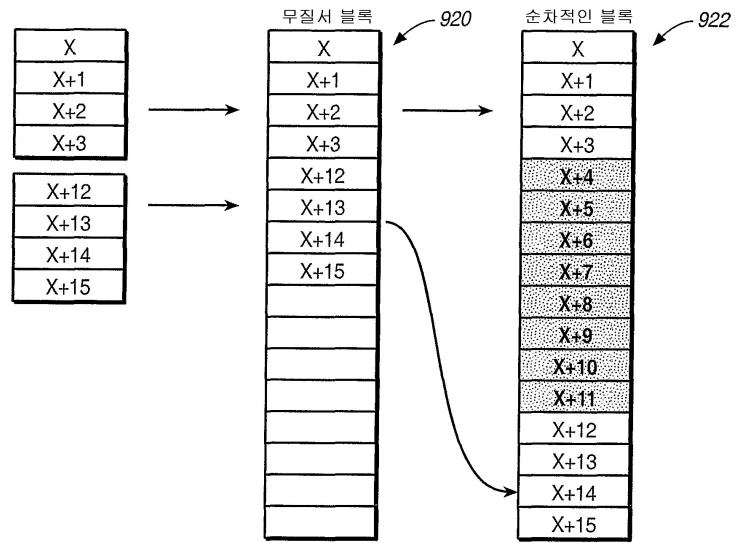
도면9b



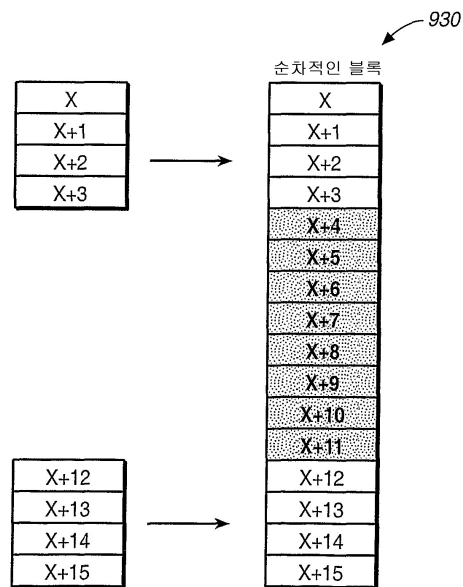
도면9c



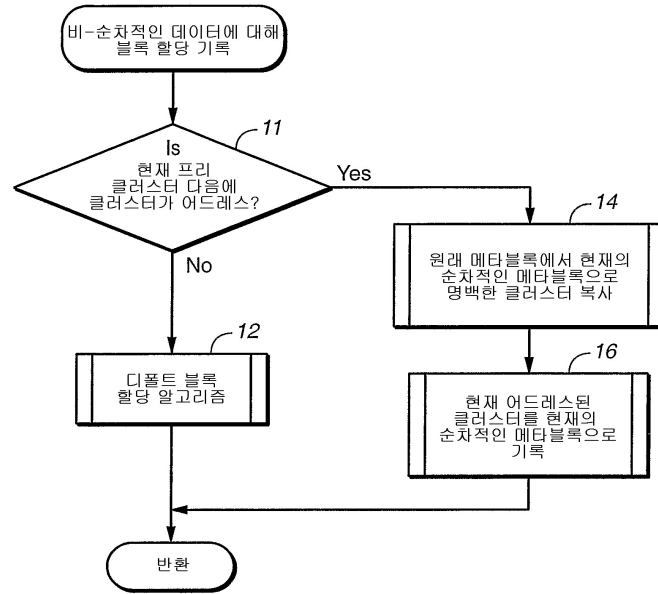
도면9d



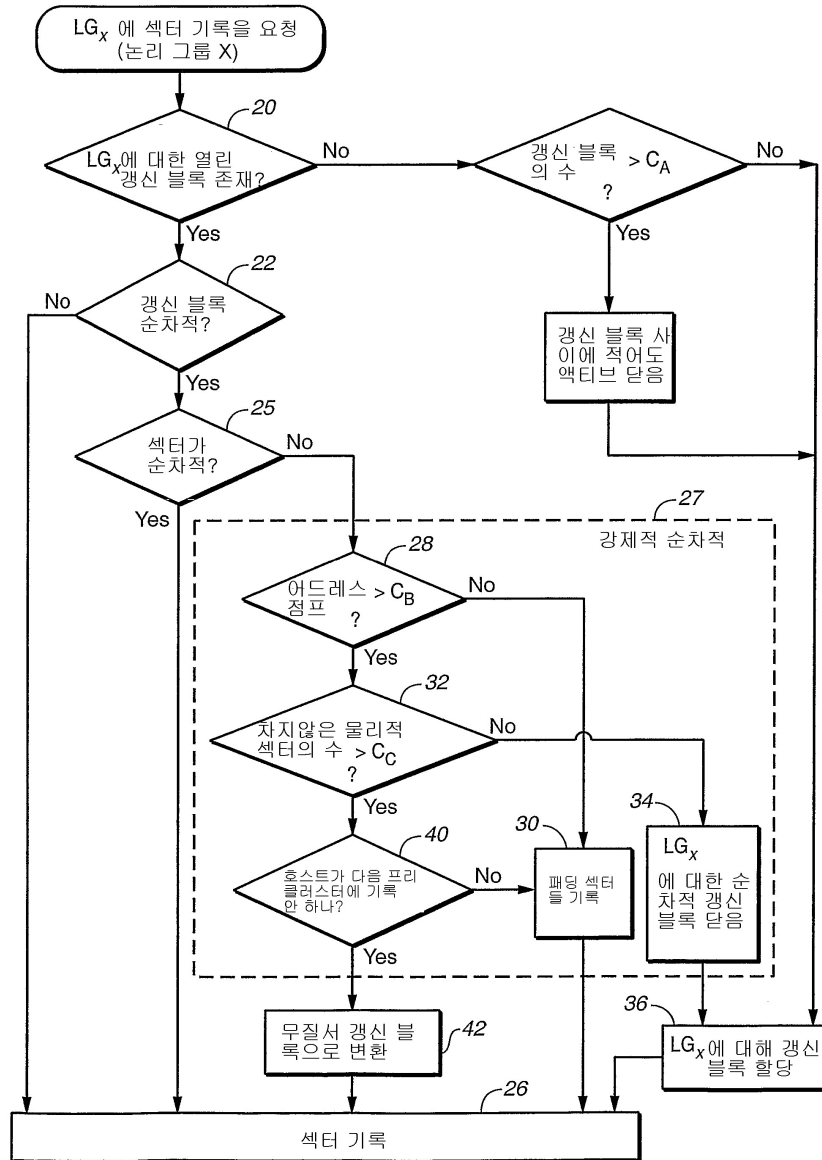
도면9e



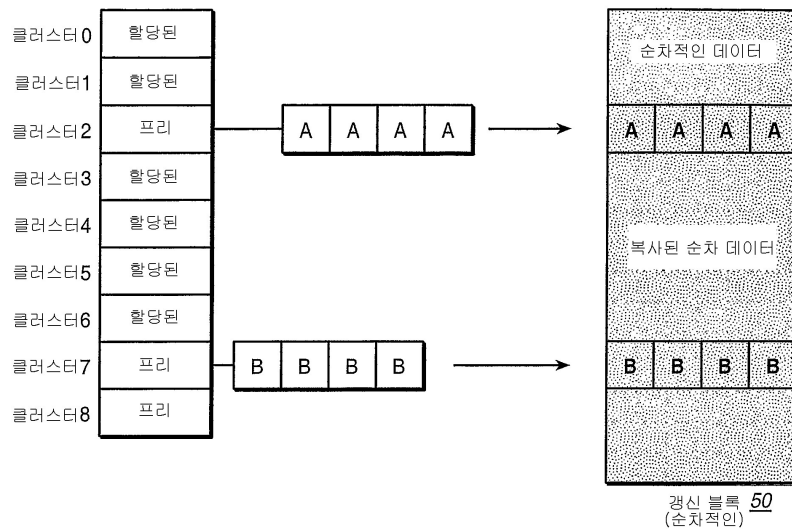
도면10



도면11

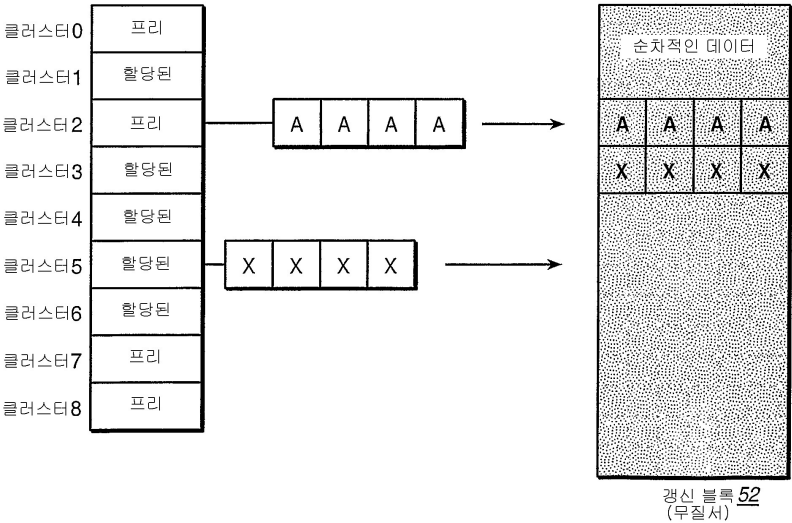


도면12a

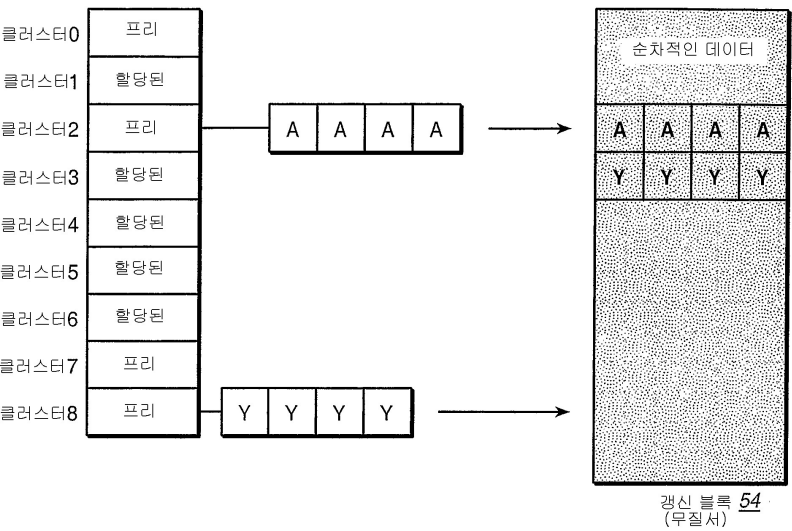




도면12b



도면12c



도면12d

