논문 2011-48CI-5-10

SSD의 성능향상을 위한 접근빈도에 따른 데이터 할당 및 교체기법

(Data allocation and Replacement Method based on The Access Frequency for Improving The Performance of SSD)

양 유 석*, 김 덕 환**

(Yu-Seok Yang and Deok-Hwan Kim)

요 약

SSD는 낸드 플래시 메모리의 배열로 구성되어 있기 때문에 하드 디스크와는 달리 블록 당 쓰기 가능 횟수가 정해져 있고, 덮어쓰기가 불가능 하다는 특성을 가지고 있다. 이와 같이 기존의 하드 디스크와는 다른 특징을 갖는 SSD를 효과적으로 관리하기 위해 FTL을 이용한다. FTL은 맵핑 방식에 따라 페이지, 블록, 로그 블록 맵핑 방식으로 구분하는데, 그 중 로그 블록 맵핑 방식 기법 중 BAST와 FAST는 합병 연산 시 페이지 복사와 삭제 연산이 많이 발생하여 SSD의 성능이 떨어지는 문제를 갖고 있다. 본 논문에서는 이를 해결하기 위하여 SSD 내부에 PRAM을 접근빈도 체크영역 및 로그 블록과 Hot 데이터를 저장하는 영역으로 할당하여 접근빈도에 따라 데이터를 할당하는 기법 및 교체기법을 제안한다. 제안된 방법은 접근빈도에 따라 Cold 데이터는 플래시 메모리에 할당하며 덮어쓰기가 가능한 PRAM에 로그 블록과 접근 빈도가 높은 데이터를 할당함으로써, 합병 연산 및 삭제 연산을 줄여 SSD의 성능과 수명을 향상시킬 수 있다. 또한 용량의 한계가 있는 PRAM의 활용률을 높이기 위해 데이터 교체 방법을 사용한다. 실험결과 삭제연산의 경우 제안한 방법이 BAST에 비해 약 46%정도 FAST에 비해 약 38%정도 횟수가 감소되었고, 쓰기 성능의 경우 각각 기존 BAST, FAST에 비해 34%, 19%의 성능이 향상되었고, 읽기 성능 역시 각각 5%, 3%의 성능 향상을 보였다.

Abstract

SSD has a limitation of number of erase/write cycles and does not allow in-place update unlike the hard disk because SSD is composed of an array of NAND flash memory. Thus, FTL is used to effectively manage SSD of having different characteristics from traditional disk. FTL has page, block, log-block mapping method. Among then, when log-block mapping method such as BAST and FAST is used, the performance of SSD is degraded because frequent merge operations cause lots of pages to be copied and deleted. This paper proposes a data allocation and replacement method based on access frequency by allocating PRAM as checking area of access frequency, log blocks, storing region of hot data in SSD. The proposed method can enhance the performance and lifetime of SSD by storing cold data to flash memory and storing log blocks and frequently accessed data to PRAM and then reducing merge and erase operations. Besides, a data replacement method is used to increase utilization of PRAM which has limitation of capacity. The experimental results show that the ratio of erase operations of the proposed method is 46%, 38% smaller than those of BAST and FAST and the write performance of the proposed method is 34%, 19% higher than those of BAST and FAST, and the read performance of the proposed method is 5%, 3% higher than those of BAST and FAST, respectively.

Keywords : SSD, Flash memory, FTL, PRAM, Data allocation

* 학생회원, ** 정회원, 인하대학교 전자공학과
(Department of Electronic Engineering, Inha University)
※ 본 논문은 지식경제부와 한국산업기술진흥원의 전략기술인력양성사업 및 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업(2011-0004114), 그리고 2011년도 정부(교육과학기술부)의 재원으로 한국연구재단의 중점연구소 지원사업(2011-0018394)으로 수행된 연구임.
접수일자: 2011년5월3일, 수정완료일: 2011년8월30일

Ⅰ. 서 론

SSD는 낸드 플래시의 배열로 구성되어 있기 때문에 전력 소모가 적고, 충격에 강하며, 전원 공급이 중단 되어도 정보가 사라지지 않는 비휘발성인 특징을 가지고 있다. 하지만 플래시 메모리는 기존의 하드 디스크와는 다른 두 가지의 특성을 가지고 있다.

첫째, 플래시 메모리의 각 블록 당 쓰기 가능 횟수가 제한되어져 있다. 현재 플래시 메모리는 SLC (Single Level Cell) 타입과 MLC (Multi Level Cell) 타입으로 구분할 수 있는데, SLC의 경우 약 10만 번의 쓰기가 가능하고, MLC의 경우 SLC의 1/10인 약 1만번 정도의 쓰기가 가능 하다. 공간 효율성을 높이기 위하여 MLC 타입 이상의 메모리가 사용되게 될 경우 더 낮은 수명을 가지게 될 것이기 때문에 효율적인 데이터 균등 방식이 필요하다.

둘째, 플래시 메모리는 일반 디스크와는 다르게 덮어쓰기가 불가 하여 데이터를 쓰기 전에 지워야 하는 문제가 있다. 덮어쓰기가 안 되는 특징으로 인해 삭제 연산이 많이 일어나게 되어 저장장치의 수명에 영향을 미치며, 느린 삭제 연산 속도로 인해 성능도 떨어지게 된다.

이처럼 기존의 일반 디스크와는 다른 특징들을 가지고 있다. 이러한 플래시 메모리의 여러 특성을 효과적으로 관리하기 위해 FTL(Flash Translation Layer)[1]을 이용하여, 블록 디바이스 설정에 관계없이 일반 디스크처럼 플래시 메모리 저장장치를 관리한다.

FTL은 맵핑 방식에 따라 세 가지로 분류 하는데, 페이지 단위로 맵핑하는 페이지 맵핑 방식[2], 블록 단위를 기준으로 하는 블록 맵핑 방식[3], 그리고 페이지 맵핑 방식과 블록 맵핑 방식의 장점을 취한 하이브리드 맵핑 방식이 있다[4~5]. 페이지 맵핑 방식의 경우 쓰기의 가장 작은 단위인 페이지를 기준으로 하기 때문에, 플래시 메모리의 저장 공간이 클수록 맵 테이블의 크기가 커지게 되고, 맵 테이블을 관리하는 DRAM의 저장 공간 또한 늘어나야 하는 단점을 가지고 있다. 맵 테이블의 크기를 줄이기 위해 페이지를 묶은 단위인 블록을 기준으로 맵 테이블을 만드는 블록 맵핑 방식은 맵 테이블의 크기를 줄일 수 는 있지만, 저장 공간 효율이 낮고, 삭제 연산이 많이 일어난다는 단점이 있다. 이 두 맵핑 방식의 장점만을 취한 하이브리드 맵핑 또는 로그 블록 맵핑 방식은 합병연산 시 많은 페이지 복사와 삭제 연산이 발생하는 단점이 있다.

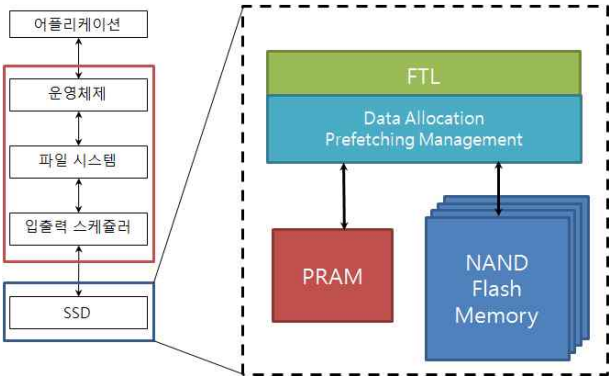


그림 1. 제안하는 SSD의 구성도
Fig. 1. The proposed Structure of the SSD.

본 논문에서는 이러한 낸드 플래시 메모리의 제한된 쓰기 가능 횟수 문제와 삭제연산으로 인한 성능 감소문제를 해결하기 위해, 접근빈도에 따른 데이터 할당기법[6]과 PRAM (Phase-change Random Access Memory)[7~8]을 접근빈도 체크영역 및 로그 블록과 Hot 데이터를 저장하는 영역으로 사용하여 SSD의 성능을 향상시키는 기법을 제안한다. 논문에서 주장하는 SSD의 구성도는 그림 1과 같다[9].

플래시 메모리는 블록 맵핑 방식을, PRAM은 페이지 맵핑 방식을 사용하고, 하이브리드 맵핑 방식에서 쓰기 버퍼로 사용하던 로그 블록을 플래시 메모리가 아닌 덮어쓰기가 가능한 PRAM에 저장하는 방법을 사용한다. 또한 접근 빈도를 체크하여, 접근 빈도가 높은 데이터를 Hot 데이터, 접근 빈도가 낮은 데이터를 Cold 데이터로 구분하고 Hot 데이터를 PRAM으로 할당하여 쓰기/삭제 연산을 줄이고 성능을 향상시키는 방법을 제안한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 로그 블록 맵핑 방식중 하나인 BAST와 FAST방식과, 차세대 NVRAM인 PRAM에 관해 기술 한다. Ⅲ장에서는 본 논문의 주요 내용인 데이터 할당 기법과 데이터 교체 방법을 기술하고, Ⅳ장에서는 실험을 통해 제안한 방법의 신뢰성 및 실험결과를 보인다. 마지막으로 Ⅴ장에서는 결론을 맺는다.

Ⅱ. 관련 연구

낸드 플래시는 덮어쓰기가 안 된다는 단점을 가지고 있다. 따라서 덮어쓰기를 하려면, 쓰기 요청이 있을 때

마다 해당 요청 페이지가 있는 블록을 전부 지우고 새로 쓰는 방법을 사용할 수는 있지만, 이 방법은 메모리의 수명 측면에서 문제가 된다. 이러한 문제점을 보안하기 위해 맵핑 방식을 사용하게 되었다.

본 절에서는 하이브리드 맵핑 방식 중 BAST와 FAST에 대해 설명 하고, 데이터 할당 기법을 사용하기 위한 PRAM에 대해 설명한다.

1. BAST (Block Associative Sector Translation)

BAST는 로그 블록이라는 빈 블록을 쓰기 버퍼로 할당하여 쓰기 요청을 처리하는 방식을 사용한다. 데이터 블록과 로그 블록은 1:1로 연관 되어 있으며, 데이터 블록에서 갱신 연산이 발생할 경우, 로그 블록이 할당되어 로그 블록의 페이지에 데이터를 기록한다. 로그 블록은 데이터 블록에 비해 적은 숫자를 할당하였기 때문에 로그 블록이 전부 사용되어지면, 비어있는 블록을 만들기 위해 로그 블록에 저장된 데이터들을 다시 데이터 블록으로 전송하고, 삭제하는 방법을 사용하였다.

그림 2는 BAST 기법에서 로그 블록에 데이터를 쓰는 과정을 보여준다. 그림의 특징은 데이터 블록과 로그 블록이 1:1로 대응된다는 점이다. BAST는 순차적인 데이터 갱신에 좋은 성능을 보이나, 랜덤쓰기가 발생할 경우 로그 블록이 전부 사용되기 전에 새로운 로그 블록을 할당해야 하는 문제가 발생한다.

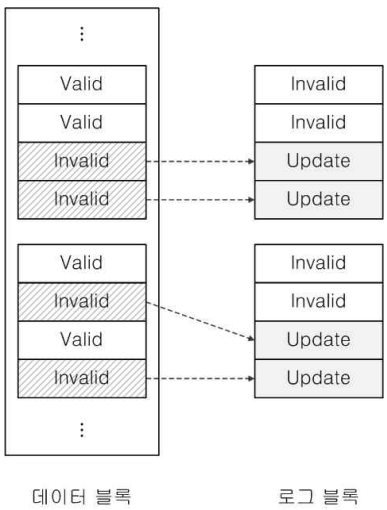


그림 2. BAST 기법
Fig. 2. BAST scheme.

2. FAST (Fully Associative Sector Translation)

FAST는 BAST의 단점인 로그블록 활용률을 높이기 위해 모든 데이터 블록들이 로그블록을 공유하는 방법을 제안하였다. FAST는 로그 블록과 데이터 블록의 연관관계를 1:N으로 확장하고, 로그 블록을 순차 쓰기와 임의 쓰기를 구분하여 처리하기 위한 순차쓰기용 로그 블록과 임의 쓰기용 로그 블록을 따로 관리하여 처리함으로써 로그블록의 활용율을 높였다. 따라서 같은 작업 환경에서 BAST보다 합병연산이 적게 발생하고 불필요한 읽기/쓰기/삭제 연산이 줄어들었다. 하지만 FAST는 임의 쓰기와 순차 쓰기 명령이 혼합되어 들어오는 경우에는 효율적으로 처리하지 못하는 단점을 가지고 있다.

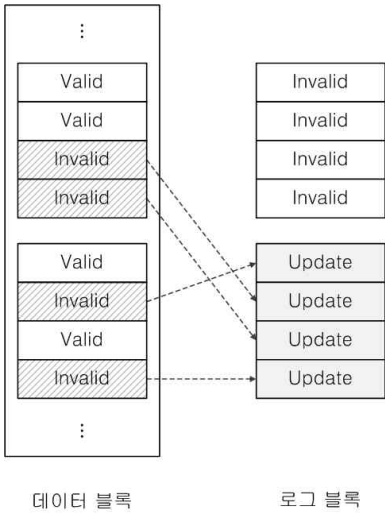


그림 3. FAST 기법
Fig. 3. FAST Scheme.

3. PRAM
(Phase-change Random Access Memory)

PRAM은 상변화에 따른 저항의 차이로 데이터를 저장하는 메모리 소자로서, 현재 PRAM은 주로 DRAM과 CPU의 캐쉬를 대체하기 위한 메모리로도 연구되고 있다[10]. PRAM을 사용하여 메인메모리나 CPU의 캐쉬를 사용할 경우, 비휘발성의 특징을 이용해 운영체재의 부팅 시간을 단축 할 수 있다. 이 장점은 일반 컴퓨터 환경에서 큰 장점이 아닐 수 있지만, 모바일 환경의 경우 빠른 부팅과 비휘발성의 장점으로 인해 NOR 플래시를 대체 할 수 있어 많은 연구가 진행 되고 있다[11~13]. PRAM의 경우 비휘발성이라는 장점이 있지만, DRAM이나 SRAM에 비해 속도가 떨어지기 때문에 실제 적용하는데 어려움이 있다. 하지만 저장장치로는 플래시 메모리보다 저장 속도가 빠르고, 플래시 메모리의 덮어쓰기가 안 되는 단점을 보안할 수 있어 플래시 메모리를

표 1. MLC 낸드플래시와 PRAM의 성능 비교
Table 1. Performance comparison of MLC NAND flash and PRAM.

Memory device	Access time		
	Read	Write	Erase
MLC NAND	121us (page)	861us (page)	1.5ms (block)
PRAM	80ns (word)	10us (word)	N/A

대체 할 수 있는 차세대 저장장치로 주목 받고 있다. 따라서 본 논문에서는 PRAM을 사용하여 낸드 플래시의 단점을 보안하고, 성능을 향상 시키는 방법을 제안한다.

표 1은 MLC타입 낸드 플래시 메모리와 PRAM의 성능 비교표 이다[11].

Ⅲ. 본 론

본 장에서는 접근 빈도에 따른 데이터 할당 기법과, PRAM을 이용하여 접근빈도를 체크하고, Hot 데이터를 저장하는 방법을 기술한다. 그리고 용량이 한정적인 PRAM에서 플래시메모리로의 데이터 교체 방법에 대해 기술한다.

1. 데이터 할당기법

일반적으로 요청되는 데이터는 쓰기보다 읽기가 많다. 따라서 플래시 메모리 보다 읽기 속도가 높은 PRAM을 사용할 경우 성능 향상을 가져올 수 있다. 그리고 접근 빈도가 높은 데이터 일수록 자주 사용될 확률이 높기 때문에 데이터들을 덮어쓰기가 가능한 PRAM에 저장함으로써 데이터 갱신 횟수를 줄 일 수 있다.

본 논문에서는 데이터의 갱신 빈도수를 고려하여, 데이터 영역을 Hot과 Cold로 구분하고, 플래시 메모리에 읽기/쓰기 요청이 들어오는 경우, 데이터의 접근 빈도수를 체크하여 데이터를 분류하는 방법을 사용한다.

본 논문에서 제안하는 하이브리드 맵핑 방법은 그림 4와 같다.

플래시 메모리는 블록 맵핑 방법을 사용하여 관리하고, PRAM의 경우 페이지 맵핑 방법을 사용하여 관리한다. 일반적인 로그 블록 맵핑 기법을 사용하며, 플래시 메모리의 일정 부분을 로그 블록으로 사용하던 기존의 방식 대신에, 로그 블록을 PRAM에 저장하는 방식을 사용한다.

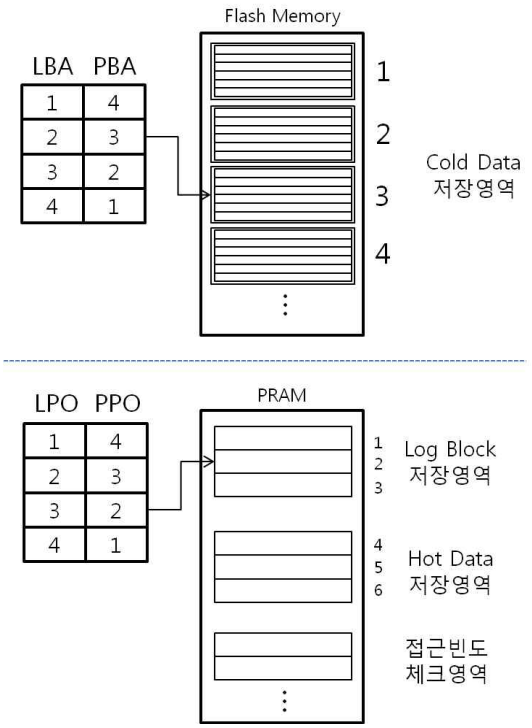


그림 4. 제안하는 맵핑 방법
Fig. 4. The proposed mapping method.

로그 블록을 PRAM에 저장함으로써 발생하는 이점은 다음과 같다.

첫째, 삭제 연산를 줄여 플래시 메모리의 신뢰성을 높인다. 플래시 메모리는 지우는 횟수에 제한을 가지고 있다. 기존의 하이브리드 맵핑 방법 중 하나인 BAST의 경우, 데이터 블록의 수에 비해 로그 블록의 수는 적고, 한 개의 데이터 블록에 한 개의 로그블록을 할당하기 때문에 여러 데이터 블록에 대해 덮어쓰기 연산이 발생할 경우 잦은 합병연산이 발생하여, 마모도에 많은 영향을 미친다. 그에 반해 PRAM의 경우 덮어쓰기 연산이 가능한 이점을 가지고 있기 때문에 로그 블록을 전부 활용할 수 있다. 따라서 상대적으로 비용 부담이 큰 완전 합병 연산의 수를 줄일 수 있고, 그에 따른 삭제 연산 역시 줄일 수 있어서, 플래시 메모리의 신뢰성을 향상 시킬 수 있다.

둘째, 낸드 플래시 보다 빠른 접근속도의 이득을 얻을 수 있다. PRAM은 플래시 메모리에 비해 접근속도가 더 빠르다는 이점을 가지고 있다. 따라서 플래시 메모리를 로그 블록으로 사용했을 때 보다 빠른 성능을 얻을 수 있다. PRAM에 Hot 데이터를 저장하는 할당 방식도 이러한 이점을 이용하고 있다.

그림 5는 전체 PRAM을 사용한 플래시 저장장치의

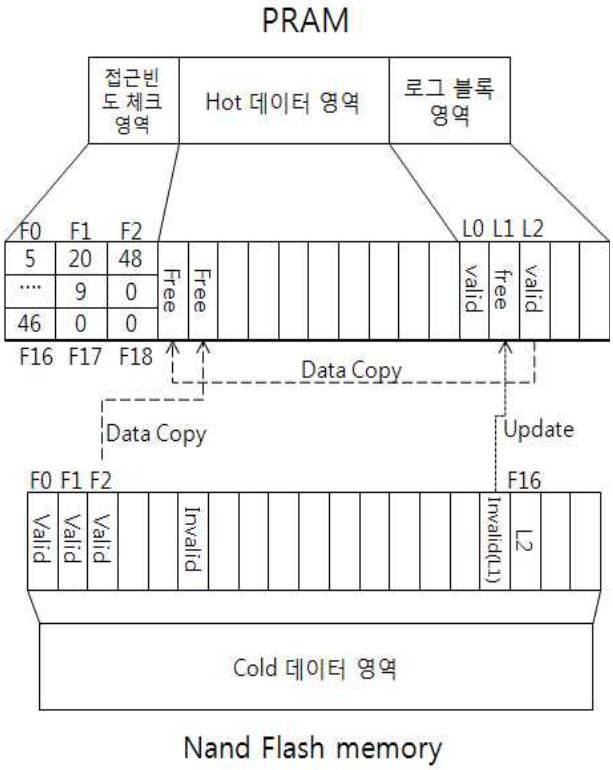


그림 5. 제안하는 저장장치 전체 구조
Fig. 5. The proposed storage device, the entire structure.

전체 구조를 보여준다. 플래시 메모리는 Cold 데이터 블록 영역으로, PRAM을 접근빈도 체크영역 및 로그 블록과 Hot 데이터를 저장하는 영역으로 사용한다.

그림 6은 PRAM을 이용한 데이터 할당 기법의 구성에 대한 설명을 위한 그림이다.

예를 들어 그림 6과 같이 읽기/쓰기 요청이 5, 7, 7, 5 순서로 들어올 경우, 순서대로 로그 블록에 기록이 되고 맵핑 테이블과 PRAM의 일부 영역을 사용하는 접근빈도수 체크 영역을 갱신 한다. 접근 빈도수가 많다는 것은 그만큼 많이 사용된다는 것이므로, 접근 빈도수가 지정된 한계치를 넘어가게 되면 그 블록 Hot 데이터로 분류되어 그림과 같이 PRAM의 비어있는 Hot 데이터 블록을 할당하여 데이터를 이동한 후 실제 블록 주소의 위치를 할당된 Hot 데이터 블록의 위치로 변경한다.

PRAM에 Hot 데이터를 할당함으로써 얻는 이점은 로그블록을 PRAM에 저장함으로써 얻는 이점과 비슷하다. 우선 접근 빈도 횟수가 많은 데이터일수록 임의적이고 작은 데이터이며, 다음 읽기/쓰기 요청시 사용될 확률이 높다. 플래시 메모리는 임의 쓰기, 읽기가 연속적인 읽기 쓰기보다 상대적으로 느린 단점을 가지고

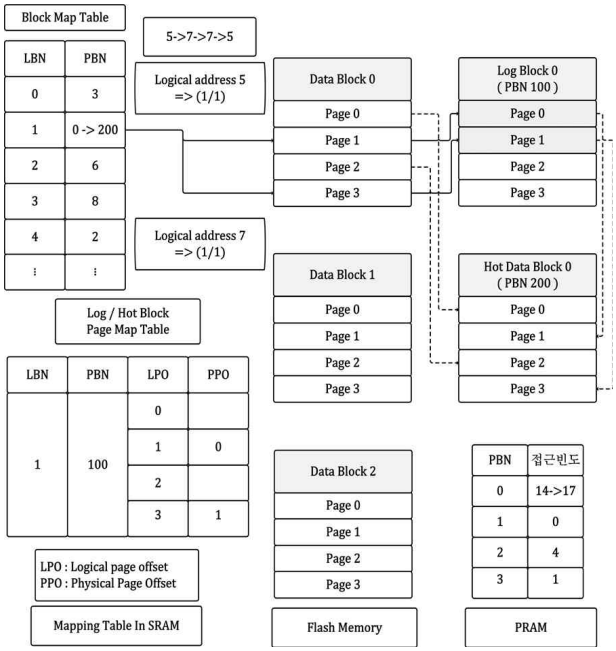


그림 6. PRAM을 이용한 데이터 할당 방법
Fig. 6. Data allocation method using PRAM.

있는데 플래시 메모리보다 속도가 빠른 PRAM을 사용하여 이러한 단점을 보안 하여 성능을 향상 시킬 수 있다. 또한, 데이터 업데이트 시, 플래시 메모리를 거치지 않고, 덮어쓰기가 가능한 PRAM에서 처리되기 때문에, 플래시 메모리의 삭제 연산을 줄일 수 있고, 그에 따른 플래시 메모리의 마모도를 줄일 수 있어 저장장치의 신뢰성을 향상 시킬 수 있다.

2. 데이터 교체 방법

로그 블록과, Hot데이터 할당 영역으로 사용되는 PRAM의 용량에 한계가 있다. 따라서 PRAM에 할당된 Hot 데이터블록과 플래시 메모리의 Cold 데이터 블록의 교체 연산이 필요하다. 데이터의 접근 빈도를 체크하기 위해 PRAM의 일부 영역을 사용하여 접근 빈도를 카운트 한다. 접근 빈도수 체크 영역의 크기는, 맵핑 테이블의 크기를 계산하는 방식과 같이 방식과 같이 플래시메모리와 PRAM의 블록의 수를 기준으로 계산된다.

접근 빈도수를 체크하여 Hot 데이터에서 Cold 데이터의 임계값으로 떨어진 PRAM의 데이터 블록과 Cold 데이터에서 Hot 데이터의 임계값으로 올라간 플래시 메모리의 데이터 블록을 교체 한다.

데이터 교체 알고리즘의 순서도는 그림 7과 같다.

먼저 입/출력 요청이 들어오면 PRAM의 접근빈도 체크 영역을 이용하여 접근빈도수를 체크한다. 만약 접근

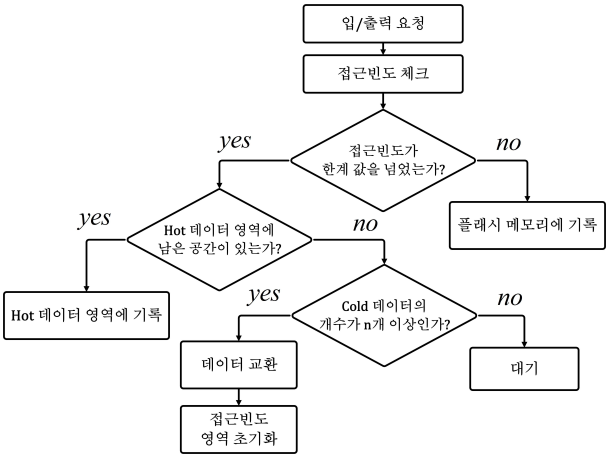


그림 7. 데이터 교체 순서도
Fig. 7. Data replacement flow chart.

빈도 수가 정한 한계 값을 넘지 않았을 경우나 처음 들어온 요청일 경우는 플래시 메모리에 저장 하고 접근빈도 수를 올린다. 한계 값을 넘었을 경우, Hot 데이터 영역에 남은 공간이 있는지 살펴보고 공간이 남아 있다면 Hot 데이터 영역에 데이터 복사를 하고, 남은 공간이 없을 경우 Hot 데이터와 Cold 데이터의 교체 정책을 실시한다.

교체 연산시 데이터들이 자주 이동하게 될 경우 오히려 성능 저하를 가져 올 수 있기 때문에, Hot데이터 영역에 접근 빈도수가 낮아진 데이터를 체크하여 그 블록의 개수가 N개 이상일 경우 데이터 교환 정책을 실시한다. N개가 넘지 않았을 경우 N개 이상이 될 때까지 데이터 교체를 연기한다. 만약 N값을 넘었을 경우 데이터 교체가 일어나고 PRAM의 일부 영역을 사용하는 접근 빈도 체크 영역을 초기화하여 다시 접근빈도에 따라 Hot 데이터와 Cold 데이터를 구분한다.

접근빈도의 한계 값과 Hot 데이터 영역의 Cold 데이터 개수 N의 값은 실험결과를 통해 알아본다.

Ⅳ. 실험 및 성능 평가

본 논문의 실험은 낸드 플래시 메모리와 PRAM의 혼합기반 저장장치에서 데이터 할당기법을 사용하여 데이터를 구분하고, 접근 빈도가 높은 데이터를 PRAM에 할당 하는 방법을 사용한다. 다음 실험을 통해 기존 하이브리드 맵핑 방식과 삭제횟수, 마모도 및 읽기/쓰기 성능을 비교한다.

1. 실험 환경

표 2는 사용된 실험 환경을 보여준다. SSD를 구성하는 플래시메모리 와 PRAM은 Disksim 기반의 SSD 모듈이 패치된 시뮬레이터[14]를 사용하여 구현 하였으며, 실험 데이터들은 리눅스 환경에서 일반적인 PC 환경에서의 프로그램 설치, 동영상 재생, 게임, 웹서핑 등의 상황을 blktrace 프로그램을 이용하여, trace 파일을 추출하여 실험하였다. 비교 대상으로 하이브리드 맵핑 방식 중 BAST 기법과 FAST 기법을 사용 하였다.

표 2. 실험 환경
Table 2. Experimental Environment.

항목	시스템 사양
운영체제	Linux kernel 2.6.29
CPU	Intel core 2 duo (2.4Ghz)
Memory	DDR2 1GB * 3개
저장장치	MLC NAND 16GB
	PRAM 64MB

2. 접근빈도 한계 값과 데이터교환을 위한 N값

Hot 데이터와 Cold데이터를 구분하기 위해 접근빈도의 임계값을 설정하고 데이터 교체 횟수를 줄이기 위해 N값을 예측할 필요가 있다. 그림 8은 접근빈도의 임계값과 N값에 따른 실행시간을 보여준다.

실행시간을 기준으로 실험한 결과 실험데이터에 따

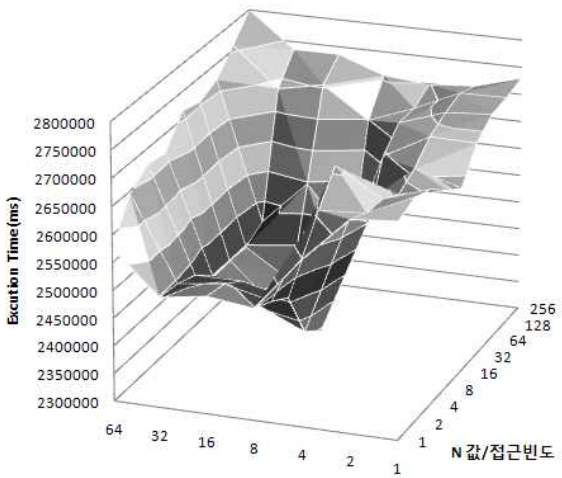


그림 8. 접근빈도와 N값에 따른 실행시간 비교
Fig. 8. Comparison of execution time and access frequency of the N.

라 조금씩 차이를 보이나 Hot 데이터로 분류하기 위한 접근빈도 한계 값은 16에서 데이터 교환 정책을 위한 N값은 8일 때 각각 가장 좋은 성능을 보여주었다.

3. 로그 블록 수에 따른 삭제 및 전체 삭제 횟수 분석

그림 9는 로그 블록의 개수를 변경하여, 소거횟수의 차이를 비교한 실험결과이다.

로그 블록의 수가 가장 적은 4개로 설정 하였을 때, 소거 연산이 가장 많이 발생 하였고, 로그 블록의 수가 커질수록 소거 연산이 줄어드는 것을 볼 수 있다. 제안한 기법은 BAST 방식에 비해 60%에서 40% FAST 방식에 비해 80%에서 60% 정도의 적은 소거 횟수를 보였다.

로그 블록이 많을수록 합병 연산을 지연 시킬 수 있는 장점이 있지만, 로그 블록의 수가 많아지면 그로 인해 Hot 데이터를 저장할 수 있는 공간의 활용률이 떨어지는 단점이 있다.

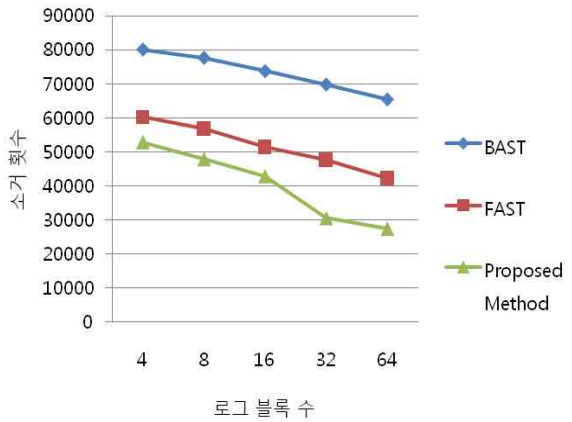


그림 9. 로그 블록 수에 따른 소거횟수 차이 비교
Fig. 9. Compare the difference in the number of erase cycles according to the log block.

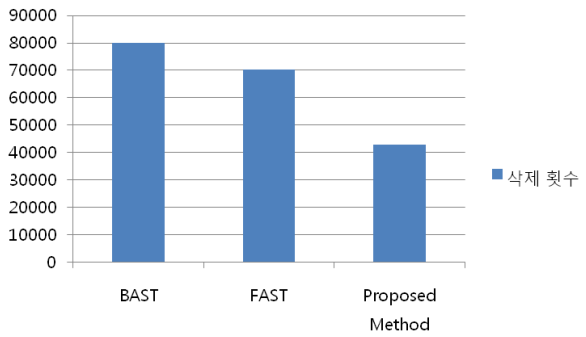


그림 10. 삭제연산 횟수의 비교
Fig. 10. Comparison of erase operation.

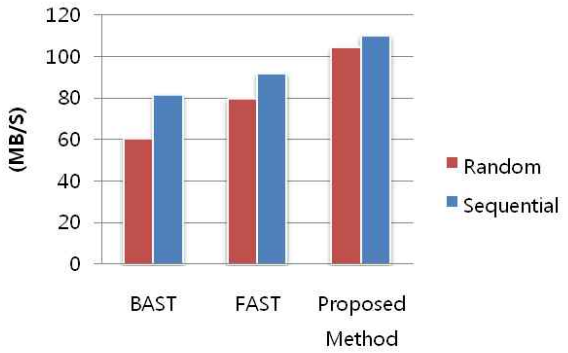
그림 10은 전체 삭제연산 횟수를 비교한 결과이다.

제안한 방법은 접근빈도가 높은 데이터와 로그블록을 덮어쓰기가 가능한 PRAM에 저장하는 방법을 사용하였다. 따라서 덮어쓰기가 안 되는 플래시메모리를 사용한 방법보다 합병의 수를 줄일 수 있었고, 그 결과로 제안한 방법이 BAST 보다 약 46%, FAST 보다 약 38%정도 삭제연산 횟수가 감소하였다.

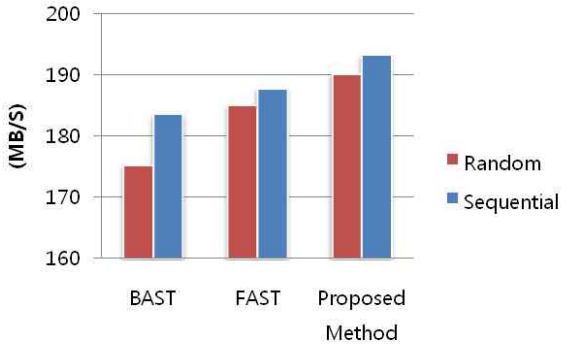
4. 제안된 구조에 따른 쓰기/읽기 성능 분석

그림 11은 순차와 임의 쓰기 및 읽기 성능을 나타낸다.

쓰기 성능의 경우 플래시메모리에 비해 빠른 PRAM의 쓰기속도 차이와 로그 블록과 Hot 데이터 영역의 영향으로 합병 연산이 줄어 순차 쓰기 방식의 경우 BAST에 비해 약 34% FAST에 비해 약 19%, 임의 쓰기의 경우 각각 59%, 31%의 성능 향상을 보여주었다. 읽기 성능의 경우 접근빈도가 높은 Hot 데이터를 분류하여 읽기 성능이 좋은 PRAM에 할당함으로써 얻어지



(a) 쓰기 성능



(b) 읽기 성능

그림 11. 쓰기/읽기 성능 비교
Fig. 11. Write / read performance compared.

는 접근속도 차이로 인해 순차 읽기의 경우 BAST에 비해 약 5% FAST에 비해 3%, 임의 쓰기의 경우 역시 각각 8%, 3% 정도의 성능향상을 보였다.

Ⅳ. 결 론

본 논문에서는 SSD의 덮어쓰기가 불가능 하여 쓰기 전에 지워야함으로 생기는 블록 당 쓰기 가능 횟수 제한 문제와, 읽기/쓰기 성능을 향상시키기 위하여 ,데이터 할당기법 및 교체기법과 PRAM을 접근빈도 체크영역 및 로그 블록과 Hot 데이터를 저장하는 영역으로 사용하는 방법을 제안하였다.

제시된 기법은 하이브리드 맵핑 FTL을 기반으로 하고 있으며 접근빈도가 낮은 Cold 데이터는 플래시메모리에, 로그 블록과 접근빈도가 높은 Hot 데이터를 덮어쓰기가 가능한 PRAM에 할당함으로써, 합병 연산 및 삭제연산을 줄여 SSD의 마모도를 줄일 수 있었고, 플래시 메모리보다 성능이 좋은 PRAM의 영향으로 기존의 BAST 방식과 FAST 방식에 비해 삭제연산의 수가 감소되었고, 읽기, 쓰기 성능이 향상되었음을 확인 할 수 있었다.

참 고 문 헌

[1] Intel Corporation, “Understanding the Flash Translation Layer Specification,” http://developer.intel.com, 1998.

[2] A. Birrel, M. Isard, C. Thancker, and T. Wobber, 2007. “A design for High-Performance Flash Disks”. ACM SIGOPS Operating Systems Review, 41(2), April 2007.

[3] A. Ban. “Flash file system. United States Patent”, No. 5,404,485, April 1995.

[4] J. Kim, J.M. Kim, S.H. Noh, S. Min, and Y. Cho. “A Space-Efficient Flash Translation Layer for Compactflash Systems”, IEEE Transactions on Consumer Electronics, 48(2), pp.366-375, 2002.

[5] S.-W. Lee, D.-J. Park, T.-S. Chung, D.H. Lee, S. Park, and H.-J. Song, “A Log Buffer-Based Flash Translation Layer Using Fully-Associative Sector Translation”, ACM Transactions on Embedded Computing Systems, 6(3), Article 18, 2007.

[6] S. Im and D. Shin. Storage architecture and software support for SLC/MLC combined flash memory. In Proceedings of the 2009 ACM symposium on Applied Computing, pp.1664-1669, 2009.

[7] G.H. Koh and et el., 2004. “PRAM Process Technology”, in Proceeding of the IEEE International Conference on Integrated Circuit Design and Technology, 2004.

[8] Kinam Kim and G.H. Koh, 2004. “Future Memory Technology including Emerging New Memories”, in Proceedings of 24th International Conference on Microelectronics, NIS, Serbia and Montenegro, May, 2004.

[9] 양유석, 송재석, 김덕환, “데이터 할당과 선반입 기법을 이용한 SSD 스토리지의 성능 향상”, 대한전자공학회 추계학술대회 논문집, pp.599-600, 2010년 11월.

[10] Y. Joo, et al. “Energy-and endurance-away design of phase change memory caches”, In Proceedings of DATE, 2010.

[11] J. K. Kim et al., “A PRAM and NAND flash hybrid architecture for high-performance embedded storage subsystems,” in Proceedings of 2008, pp. 31-40, 2008.

[12] G. Sun, Y. Joo, Y. Chen, D. Niu, Y. Xie, Y. Chen, and H. Li. A hybrid solid-state storage architecture for the performance, energy consumption, and lifetime improvement. In Proceedings of HPCA’10, Jan 2010.

[13] Y. Park et al., “PFFS: a scalable flash memory file system for the hybrid architecture of phase-change RAM and NAND flash,” in Proceedings of ACM SAC 2008, pp. 1498-1503, 2008.

[14] N. Agrawal and et al. “Design Tradeoffs for SSD Performance”, in Proceedings of USENIX, 2008.

저 자 소 개

양 유 석(학생회원)
2008년 순천향대학교 정보기술 공학부 학사 졸업.
2010년~현재 인하대학교 전자공학과 석사과정
<주관심분야 : 임베디드 시스템, 스토리지 시스템>

김 덕 환(정회원)-교신저자
2003년 한국과학기술원 컴퓨터 공학 박사.
2006년~현재 인하대학교 전자공학부 교수
<주관심분야 : 시각정보처리, 스토리지 시스템, 임베디드 시스템>