페이지 삭제정보를 활용하는 플래시 저장장치의 구조

(The Architecture of the Flash Memory Storage System using Page Delete Information)

정 호 영 † 박 성 민 †
(Hoyoung Jung) (Sungmin Park)
강 수 용 †† 차 재 혁 ††
(Sooyong Kang) (Jaehyuk Cha)

요 약 최근 저장장치로 하드 디스크를 대치하고 있는 플래시 메모리 저장장치는 물리적 특성이 하드디스크와 다르다. 이러한 플래시 메모리 저장장치의 성능을 향상시키기 위해 운영체제 및 파일시스템의 여러 계층에 걸쳐 다양한 연구가 진행되고 있다. 본 연구에서는 파일 삭제시 무효화되는 페이지 정보를 상위 계층에서 전달받아 이를 저장하고 활용하는 플래시 메모리 저장장치의 구조를 제안하고 해당 시스템의 성능 및 영향에 대해 연구하였다. 제안하는 시스템은 페이지 무효 정보를 블록 병합, 웨어 레벨링 등에 활용하고 이에 따라 시스템의 성능을 효과적으로 향상시키는 것으로 나타났다.

키워드 : 플래시 메모리, 저장장치, 파일 시스템, 페이지 무효

· 본 연구는 한국과학재단 특정기초연구(R01-2007-000-20649-0)지원으로 수행되었음
· 이 논문은 2009 한국컴퓨터종합학술대회에서 '페이지 삭제정보를 저장 및 활용하는 플래시 저장장치의 내부구조와 효과에 관한 연구'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 한양대학교 전자컴퓨터통신
horong@hanyang.ac.kr
syrilo@hanyang.ac.kr
†† 종신회원 : 한양대학교 컴퓨터전공 교수
sykang@hanyang.ac.kr
chajh@hanyang.ac.kr

논문접수 : 2009년 8월 13일
심사완료 : 2009년 10월 5일

Abstract Flash memory, which replaces hard disk recently, has different physical characteristics with hard disk. For the performance of flash memory based storage system, many researches over OS and file system layers has been doing. In this paper, we propose the architecture of flash memory based storage which uses information of page invalidation when file deletion occurs from upper layer. Also, we evaluate the performance of proposed system. Proposed system effectively increases IO performance by using page invalidation information to block merge and wear leveling algorithms.

Key words : flash memory, storage system, file system, page invalidation

1. 서 론

최근 플래시 메모리 저장장치는 소형 기기 뿐 아니라 노트북, PC, 워크스테이션 등에서 기존의 하드디스크를 대치하고 새로운 저장장치로 활용되고 있다.

플래시 메모리 저장장치는 물리적 특성이 하드디스크와 다르지만 현재의 대부분 운영체제와 파일 시스템 등은 이를 고려하지 않아 시스템의 성능 저하 원인이 되고 있다. 특히 플래시 메모리는 재쓰기를 위해 블록 단위의 삭제 연산이 필요하고, 이로 인해 랜덤 쓰기의 성능이 다른 IO 작업에 비해 좋지 않다.

그럼에도 불구하고 플래시 메모리가 가지는 다양한 장점들로 인해, 플래시 메모리 저장장치는 점차 그 영역을 확대하고 있으며, 플래시 메모리 저장장치의 성능을 향상시키기 위한 다양한 연구가 진행되고 있다.

최근에는 대표적인 운영체제인 리눅스와 윈도우즈 계열 운영체제에서 플래시 메모리를 위한 파일 삭제 연산을 지원하기 위한 연구가 한창 진행중이다[1,2]. 기존의 파일 삭제 연산은 삭제로 인해 무효화된 논리 블록에 대한 정보를 실제 저장장치로 전달하지 않고 파일의 메타 데이터만 수정하였다. 이러한 기존의 방식을 변형하여 실제 삭제 정보를 플래시 메모리 저장장치에 전달할 경우, 이로 인한 성능 향상을 기대할 수 있다.

실제 플래시 저장장치에서도 이를 지원하기 위한 연구가 진행중이나[3], 아직까지 명확히 내부구조나 인터페이스 등이 공개된 연구는 없다.

본 논문에서는 파일 삭제시 발생하는 무효 페이지에 대한 정보를 저장하고 활용하는 플래시 메모리 저장장치의 내부구조를 제안하고 해당 구조가 전체 시스템의 성능에 미치는 영향에 관해 기술하였다.

제안한 시스템은 파일 시스템으로부터 전달받은 페이지 무효 정보를 저장하고 이를 플래시 메모리의 삭제, 병합 등 내부 연산에 활용한 결과 기존의 플래시 메모리 저장장치와 비교하여, 시스템의 IO 성능을 약 15%

향상시키는 것으로 나타났다.

본 논문의 이후 내용은 다음과 같다. 2장에서는 본 논문의 연구의 관련 연구에 대해 기술하였다. 3장에서는 제안하는 시스템의 구조와 효과에 대해 기술하였다. 4장에서는 시뮬레이션 기반 실험 결과를 기술하고 5장에서는 결론과 향후 과제에 대해 기술하였다.

2. 관련 연구

2.1 페이지 무효 정보 전달의 필요성

일반적인 운영체제의 파일 시스템의 파일은 논리적으로 메타데이터와 데이터 페이지들의 조합으로 이루어져 있다[4]. 파일 시스템 포맷, 파일 삭제 등의 연산을 운영체제에서 수행할 경우에 시스템은 이들의 메타데이터 정보만 업데이트하며 실제 데이터 페이지들의 데이터는 여전히 남아 있게 된다. 이들 삭제된 파일들의 데이터 페이지들, 즉 더 이상 유효하지 않은 데이터 페이지들은 계속 물리적인 디스크 상에서 존재하게 된다. 이들 무효화된 데이터 페이지들이 실제로 삭제되는 시점은 차후 다른 파일이 해당 페이지를 할당받아 여기에 재쓰기가 일어나는 시점이다. 따라서 실제 물리적 저장장치 계층에서는 해당 페이지에 재쓰기가 발생하기 전까지, 이미 삭제된 파일의 기존 데이터 페이지들은 여전히 유효한 상태로 디스크에 유지되게 된다.

이렇게 파일 삭제시 무효화된 데이터 페이지들의 정보를 물리적 저장장치에 전달하지 않는 기존의 파일 시스템은 플래시 메모리 저장장치의 성능저하의 원인이 된다. 이는 플래시 메모리의 경우 재쓰기를 위해 기존 데이터들 중 유효한 페이지들을 새로운 블록에 복사하는 작업이 필요하기 때문이다. 예를 들어 아래 그림 1은 삭제된 파일 A가 1번 3번 페이지들을 포함할 경우 해당 블록의 유효한 페이지들을 복사하는 과정이다. 그림 1의 (a)에서는 삭제된 파일의 페이지 무효 정보가 없으므로, 모든 페이지들을 유효 페이지로 간주하고, 전체 페이지를 새로운 블록에 복사하게 된다. 한편 그림 1의 (b)와 같은 경우 페이지 무효화 정보를 이미 파일 시스템으로부터 전달받은 경우이므로, 현재 유효한 페이지인 0, 2, 4번 페이지들에 대해서만 복사가 일어나게 된다.

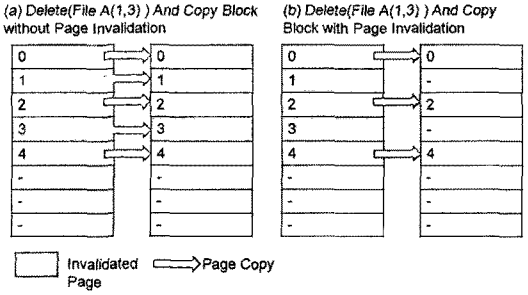

그림 1 페이지 무효가 블록 복사에 미치는 영향

그림 1과 같은 경우에서 알 수 있듯이, 플래시 저장장치에서 상위 계층으로부터 전달받은 무효 페이지 정보를 저장할 경우, 추가적인 쓰기 횟수의 감소를 통한 성능 향상을 기대할 수 있다.

또한 플래시 메모리는 각 블록의 제한된 지우기 수명으로 인해 블록간 지우기 작업의 횟수가 균등히 분배될수 있도록, 웨어 레벨링(wear leveling) 작업을 수행한다. 이 때 대부분의 웨어 레벨링 알고리즘이 고려하는 사항은 블록의 기존 지우기 작업 수행 횟수와, 블록 안의 유효한 페이지들의 개수이다. 이 때 기존의 구조에서는 파일 시스템 계층에서 무효화된 페이지들의 정보를 알 수 없으므로, 이들은 모두 유효한 페이지로 간주되고 이에 따라 의도치 않은 수행결과를 가져올 가능성을 내포하였다.

기술한 이유로 파일 시스템에서 무효 페이지들에 대한 정보를 플래시 저장장치에 전달하고, 저장장치에서 이를 활용할 경우, 시스템의 성능 및 수명 향상을 기대할 수 있다.

2.2 페이지 무효 정보를 전달하는 기존 시스템

[5] 표준에서는 최근 플래시 메모리 저장장치의 성능을 향상시키기 위해 무효 페이지 정보를 물리 장치에 전달하는 trim 연산을 제안하였으나, 아직 구체적인 방안은 확정되지 않은 상태이다. [6] 특허에서는 플래시 저장장치에 파일 삭제 정보를 전달할 수 있는 컴퓨터 시스템을 제안하고 이에 관하여 기술하고 있다. 그러나 [6] 또한 세부적인 구조에 대한 언급은 이루어지지 않고 있다. [7] 논문에서는 페이지 매핑을 사용하는 플래시 저장장치에서 삭제정보를 활용하는 방법에 대해 제안하였다. [7]에서는 파일 시스템에서 삭제로 인해 페이지 무효 정보를 플래시 저장장치로 전달할 경우, 페이지 테이블에서 해당 엔트리를 제거하여 이를 활용하는 방식을 사용하고 있으며, 이 방식은 페이지 매핑을 사용할 경우에 적합하다. 그러나 SSD와 같은 최근의 고용량 플래시 저장장치는 메모리 비용으로 인해 페이지 매핑을 사용하기 어렵고,따라서 [7]에서 제안하는 방법은 블록 매핑/ 로그 블록 매핑을 사용하는 플래시 저장장치에는 적합하지 않다.

본 논문에서는 기존의 연구를 참조하여, 파일 시스템으로부터 전달받은 페이지 무효정보를 활용할 수 있는 새로운 플래시 메모리 저장장치 구조에 대해 제안한다. 제안하는 플래시 메모리 저장장치는 페이지 매핑, 블록 매핑, 로그 블록 매핑 등 현재의 모든 매핑 기법에 쉽게 적용될 수 있으며, 추가적인 비용 또한 매우 적게 효율적으로 설계되었다. 또한 제안한 시스템의 효율성을 입증하기 위해 시뮬레이션 기반의 실험을 사용하였다.

3. 무효 페이지 정보를 저장하는 플래시 메모리 저장장치

본 논문에서 제안하는 시스템의 구조는 그림 2와 같다.

그림 2와 같이 제안하는 시스템에서는 상위 계층으로부터 전달 받은 페이지 무효정보를 플래시 저장장치 내부의 무효 페이지 테이블(Invalidation Page Table-IPT)에 저장한다. 일반적으로 무효 페이지 테이블은 매핑 테이블과 같이 콘트롤러의 Sram이나 Dram을 활용하여 저장된다.

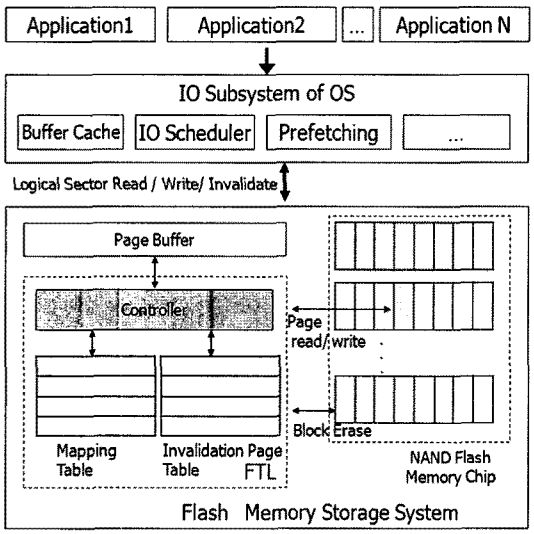

그림 2 제안 시스템의 구조

3.1 무효 페이지 테이블

그림 2의 무효 페이지 테이블에는 파일 시스템으로부터 전달받은 무효 페이지들의 정보가 저장된다. 그러나 그림 2의 FTL의 메모리는 일반적으로 그 크기가 매우 작으므로, 무효 페이지 테이블 또한 작은 크기를 효과적으로 활용해야만 한다.

따라서 제안 시스템의 무효 페이지 테이블은 다음과 같은 엔트리를 저장한다.

무효 페이지 테이블에서 엔트리의 삽입 및 갱신은 다음과 같은 규칙을 가진다. 이들 규칙은 삽입시 저장공간과 처리 비용을 최소화하는 방향으로 제안되었다.

무효 페이지 논리 번호	연속된 페이지 개수

그림 3 무효 페이지 테이블의 엔트리

1. 파일 시스템 계층에서 전달받은 무효 페이지 정보는 그림 3의 형태로 무효 페이지 테이블에 추가된다.
2. 가장 최근의 엔트리와 연속한 페이지의 경우 삽입이 아닌 최근 엔트리의 갱신이 이루어진다.
3. 최근 엔트리가 아닌 다른 엔트리에 연속된 무효 페이지의 경우 엔트리 검색 비용을 줄이기 위해 새로운 엔트리의 삽입이 일어난다.
4. 새로운 페이지에 대한 쓰기 수행시 해당 페이지에 대한 무효 정보를 포함하는 엔트리가 존재한다면 엔트리는 갱신, 또는 분할된다.

이들 엔트리는 다음과 같은 경우에는 무효 페이지 테이블에서 제거된다.

1. 블록 병합시 무효 페이지 테이블에서 검색된 페이지들은 병합시 복사대상에서 제외되며, 해당 엔트리는 무효 페이지 테이블에서 제거된다.
2. 새로운 페이지에 대한 쓰기 수행시 해당 페이지에 대한 무효 정보와 일치하는 엔트리가 존재한다면 엔트리는 삭제된다.
3. 무효 페이지 테이블에 새로운 엔트리 삽입시 공간이 부족하다면 기존 엔트리는 교체 알고리즘에 따라 삭제된다.

그림 2에서 무효 페이지 테이블의 엔트리들의 전체 크기는 시스템에서 정의한 용량을 초과할 수 없다. 만약 엔트리들이 초과될 경우는 정의된 교체 알고리즘에 따라 기존의 엔트리는 제거된다. 이 때 제거된 엔트리는 보안상의 이유 등 필요에 따라 플래시 메모리의 일부 페이지 또는 스페어 영역등에 기록될 수 있으며, 이를 기록하지 않아도 큰 영향은 끼치지 않는다.

또한 저장된 무효 페이지 테이블은 장치의 제거시 또는 시스템 종료시 플래시 메모리의 특정 공간에 백업되었다가 재시동시 이를 다시 불러 오도록 설계되었다. 따라서 이들 정보는 예기치 못한 시스템의 종료시 저장되지 못한 엔트리들에 대한 손실을 가져올 수 있으나, 이들 정보는 삭제에만 관련된 정보이므로, 시스템에 필수적인 정보는 아니다. 따라서 비정상적인 시스템 종료 후에도, 마지막으로 백업된 무효 테이블 정보만을 가져오더라도 시스템은 오류 없이 정상적으로 동작하게 된다. 다만 소실된 정보는 차후 블록 복사시 약간의 성능 감소를 가져올 수 있다.

3.2 무효 페이지 테이블 교체 알고리즘

3.1에서 제안한 무효 페이지 테이블은 블록 병합시 수행되는 유효 페이지의 복사 회수를 감소시킴으로써 성능 향상을 가져 올 수 있다. 그러나 그림 1에서 볼 수 있듯이 이로 인한 성능 향상은 블록 크기보다 작은 파일의 삭제할 경우 뚜렷하나, 블록보다 큰 대형파일의 경우에는 이득이 없게 된다. 이는 블록보다 큰 파일을 삭제할 경우 다음과 같은 과정을 거치기 때문이다.

1. 이들의 무효화 정보가 무효 페이지 테이블에 추가된다.
2. 해당 블록 전체에 다시 새로운 파일이 생성된다.
3. 새로운 파일이 생성됨으로 인해 무효 페이지 테이블 엔트리가 제거된다.

물론 대형파일로 발생한 무효 페이지의 정보들을 유지할 경우에도 웨어 레벨링시 고려하거나, 이로 인해 시스템의 유휴 시간에 빠른 삭제 연산을 수행하여, 안정적인 성능을 보장해 줄 수는 있다.

그러나 앞서 전술한 바와 같이 무효 페이지 테이블의 크기 또한 시스템에서 고려해야 할 중요한 요소이다. 무효 페이지 테이블은 작은 크기로 큰 효과를 줄 수 있어야만 한다. 따라서 무효 페이지 테이블을 위한 효과적인 교체 알고리즘이 필요하며, 본 논문에서는 관찰한 결과를 토대로 블록 크기보다 작은 페이지들의 페이지 무효화 정보를 우선적으로 교체하는 알고리즘을 제안하였다. 제안하는 알고리즘의 주요 사항은 다음과 같다.

1. 먼저 페이지에 쓰기가 발생할 경우, 무효 페이지 테이블의 엔트리들에 대한 정리 연산이 일어난다.
2. 엔트리 제거시 포함하는 연속된 페이지 수가 블록 크기보다 큰 페이지를 우선적으로 제거한다.
3. 엔트리들이 블록 크기보다 작을 경우에는 포함하는 페이지가 가장 작은 엔트리를 제거한다.

제안하는 교체 알고리즘으로 인해 본 시스템은 매우 적은 양의 메모리 비용으로도 효과적으로 동작하는 것이 실험으로 입증되었다.

4. 성능 평가

4.1 파일 병합의 개선으로 인한 성능 향상

본 절에서는 제안하는 시스템에 대한 성능을 평가하였다. 실험을 위해서 매핑 알고리즘은 [8]에서 제안한 로그블록 매핑 방법을 사용하였다.

먼저 실험을 위해서 512B~64MB까지의 크기를 갖는 임의의 파일에 대해 생성/ 사용/ 삭제를 반복하는 실험을 수행하였다. 각 파일의 접근 빈도, 수명은 그림 4와 같이 zipf 분포를 따르며, 작은 크기의 파일이 더 많이 생성, 사용, 삭제되도록 설정하였다. 파일 생성은 총 1,000,000회 반복하였으며, 이에 따라 약 5,300,000 건의 페이지쓰기 요청이 발생하였다.

그림 5는 무효 페이지 테이블의 크기에 따른 성능을 나타낸 것이다. 성능은 BAST의 수행시간을 1로 가정할 경우의 상대적인 수행시간을 나타내었다. 그림 8에서 볼 수 있듯이, 모든 삭제 정보를 저장할 경우 약 15%의 성능향상이 있는 것으로 나타났다. 또한 256개의 엔트리를

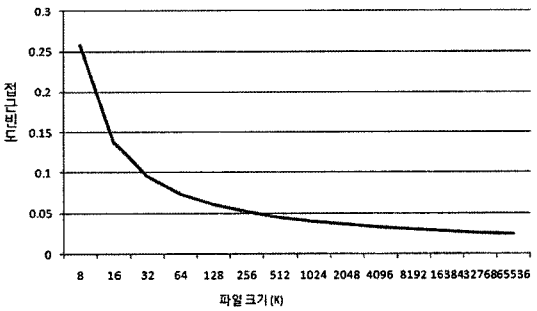

그림 4 파일 크기에 따른 접근 빈도

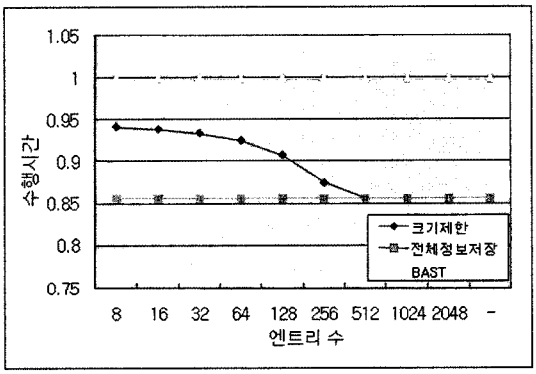

그림 5 무효 페이지 테이블 크기에 따른 성능

사용할 경우 이미 충분한 성능에 도달되는 것으로 나타났으며 512개 이상의 엔트리를 사용할 경우 최고 성능에 도달하는 것으로 나타났다. 512개의 엔트리를 저장하기 위해 필요한 용량은 전체 1,000,000회의 삭제 작업에 비해 매우 미미하며, 이는 블록 매핑 테이블의 크기보다도 매우 작은 크기이다. 따라서 제안하는 시스템이 매우 작은 양의 메모리 비용으로도 효과적으로 동작하는 것이 입증되었다.

또한 아래 표 1은 512개의 무효 테이블 엔트리를 저장하는 제안 시스템과 기존의 알고리즘의 성능을 비교한 것이다. 표에서 보는 것처럼, 무효 페이지 테이블을 사용할 경우, 많은 양의 추가 쓰기를 감소시킨 것을 볼 수 있으며, 이는 블록 병합시 무효 페이지 복사로 인한 랜덤 쓰기가 감소했기 때문으로 분석된다. 따라서 표 1에 따

표 1 무효 페이지 테이블을 사용하는 시스템의 성능

	BAST	BAST with IPT
요청	4,581,913	4,581,913
읽기	4,835,008	3,631,206
쓰기	9,672,304	8,213,120
지우기	151,094	143,114
추가쓰기비율	2.11	1.79

르면 제안하는 시스템은 플래시 메모리 저장장치의 랜덤 쓰기 성능 또한 향상시킬 수 있을 것으로 기대된다.

4.2 가비지 콜렉션 및 웨어 레벨링에 페이지 무효 연산이 미치는 영향

플래시 메모리 기반 저장장치는 불균등한 지우기로 인한 수명의 단축을 막기 위해, 가비지 콜렉션 및 웨어 레벨링 기법을 적용하고 있다[9]. 이들은 크게 블록의 지우기 횟수만을 고려하는 기법과 블록 내의 페이지 사용률을 고려하는 기법, 또는 두 가지를 모두 활용하여 적절한 블록을 선택하는 기법으로 나뉜다. 기존의 모든 알고리즘에서 블록의 페이지 사용률, 즉 블록 전체 개수에 대한 유효 페이지의 개수는 중요한 인자로 작용한다. 그러나 위의 실험과 같은 접근에서 페이지 무효 정보가 없을 경우 90% 이상의 확률로 삭제된 페이지가 유효 페이지로 간주되어 잘못된 블록을 삭제 대상으로 선택하는 경우가 발생하였다. 본 논문에서 제안하는 시스템의 경우 이러한 오류 없이, 알고리즘이 의도하는 블록을 정확히 선택하는 것이 가능해졌고, 이는 직관적으로 플래시 메모리 저장장치의 수명 및 장기적인 관점에서의 성능 향상과 직결된다는 것을 알 수 있다.

5. 결론 및 향후 과제

본 논문에서는 파일 시스템에서 전달하는 페이지 무효 정보를 효과적으로 저장하고 이를 활용하는 플래시 메모리 저장장치의 세부구조를 제안하고 제안한 시스템의 성능을 분석하였다. 시뮬레이션에 기반한 성능측정 결과 제안하는 시스템은 매우 작은 양은 메모리 비용으로 효과적으로 페이지 무효화 정보를 저장하여 시스템의 성능을 향상시키는 것으로 나타났다. 제안하는 시스템은 플래시 메모리 저장장치의 성능을 15% 향상시켰으며, 특히 랜덤 쓰기를 효과적으로 감소시킨 것으로 나타났다. 향후 과제로는 NVRAM을 활용하여, 시스템을 보다 안정적으로 개선하는 작업이 필요한 것으로 예상된다.

참 고 문 헌

[1] Satoshi Kaki, Eiichiro Sumita, and Hitoshi Iida, "A method for correcting speech recognition using the statistical features of character co-occurrence," *International Conference On Computational Linguistics*, vol.1, pp.653-657, 1998.

[2] Minwoo Jeong, Byeongchang Kim, Gary Geunbae Lee, "Semantic-oriented error correction for spoken query processing," *Automatic Speech Recognition and Understanding, IEEE*, pp.156-161, 2003.

[3] Jeffrey B. Layton: From ext3 to ext4,http://www.linux-mag.com/id/7272, 2009.

[4] Frank Sh: Windows 7 Enhancements for Solid State Drives, http://download.microsoft.com/download/5/E/6/5E66B27B-988B-4F50-AF3A-C2FF1E62180F/COR-T558_WH08.pptx, 2009.

[5] OCZ Vertex SSD New Firmware 1.10 released - Supports TRIM on Vista http://www.hardforum.com/showthread.php?t=1408940.

[6] A. Sliberschantz et al: Operating System Concepts. 6th ed., John Wiley & Sons, Inc. 2004.

[7] Frank Shu, Nathan Obrata trim command: Data Set Management Commands Proposal for ATA8-ACS2, 2007.

[8] 박찬익: 플래시 저장 장치로 삭제 정보를 전달할 수 있는 컴퓨팅시스템, 10-2008-0075707, 대한민국특허청, 2007.

[9] 김성관, 이동희, 민상렬: FAT 호환 플래시 메모리 파일 시스템을 위한 성능 최적화 기법, *한국정보과학회 2005 한국컴퓨터종합학술대회 논문집(A)*, 2005. 7.

[10] Jesung Kim, Jong Min Kim, Sam H. Noh, Sang Lyul Min and Yookun Cho: A Space-Efficient Flash Translation Layer for Compactflash systems, IEEE Trans. Consumer Electron., vol.48, no.2, pp.366-375, 2002.

[11] 이승환, 이태훈, 정기동: 플래시 메모리를 위한 페이지 비율 분석 기반의 적응적 가비지 컬렉션 정책, *한국정보과학회 2008 가을 학술발표논문집*, 제35권 제2호(A), 2008.