마모 제어 영역을 활용한 플래시 메모리 마모평준화

(Flash Memory Wear-Leveling using Regulation Pools)

박 정 수 † 민 상 렬 ††
(Jeong Su Park) (Sang Lyul Min)

요 약 본 논문에서는 마모 제어 영역(Regulation Pool)을 활용한 플래시 메모리 마모평준화 기법을 제안한다. 제안하는 기법은 FTL의 메타데이터 저장영역을 Regulation Pool로 활용하여 소거횟수가 낮은 블록에 쓰기 및 소거연산을 집중시키는 방식을 통해 간단하고 효율적인 마모평준화를 수행할 수 있다. 본 기법을 RS-FTL에 적용하여 시뮬레이션에 기반한 실험을 수행한 결과, 기법을 적용하지 않은 경우에 비해 플래시 메모리 전체 블록간 소거횟수의 편차가 약 40% 정도 감소하는 것을 확인하였다.

키워드 : 플래시 메모리, 마모평준화, FTL

Abstract In this paper, we propose a flash memory wear-leveling scheme that makes use of meta-data storage region as a regulation pool. By concentrating program and erase operations on the blocks with lower erase counts in the regulation pool, the proposed scheme achieve an even wear-leveling in a simple and efficient way. Experiments with an implementation of the proposed scheme in RS-FTL showed that the erase count deviation is reduced by around 40% through the erase count regulation.

Key words : Flash memory, Wear-leveling, FTL

· 이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2010-0015149)
· 이 논문은 2010 한국컴퓨터종합학술대회에서 'Regulation Pool을 활용한 플래시 메모리 마모평준화 기법'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 서울대학교 컴퓨터공학부
jspark.archi@gmail.com
†† 종신회원 : 서울대학교 컴퓨터공학부 교수
symin@snu.ac.kr

논문접수 : 2010년 8월 10일
심사완료 : 2010년 10월 27일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제16권 제12호(2010.12)

1. 서 론

플래시 메모리는 전원이 차단되어도 기록된 데이터가 보존되는 비 휘발성 메모리로서, 전력 소비가 적고 크기가 작으며 임의 접근 속도가 우수한 장점 등으로 인해 스마트폰, PMP 등의 소형 이동장치에서 널리 사용되고 있다. 하지만 플래시 메모리는 제자리 갱신이 불가능하고 동작 중 배드 블록이 발생할 수 있는 등 다양한 제약 사항이 존재하기 때문에 플래시 메모리의 특성을 고려한 관리기법이 필수적이다. 특히 플래시 메모리의 기본 구성단위인 블록은 소거횟수의 제약이 있으며, 이를 넘어가게 되는 경우 해당 블록은 정상적인 동작을 보장하지 못한다[1]. 이는 플래시 메모리에 대한 소거연산이 일부 블록에만 집중되는 경우 사용이 불가능해진 소수의 블록으로 인하여 전체 시스템의 수명이 단축되는 요인으로 작용한다. SLC 타입의 경우 약 100,000회, MLC 타입의 경우 약 5,000회 이내로 소거횟수가 제한되어 있으며, 플래시 메모리의 제조공정이 미세해짐에 따라 단위 블록 당 소거 가능 횟수는 점차 줄어들고 있는 추세이다[2]. 이로 인한 플래시 메모리 기반 저장 장치의 수명 문제는 더욱 심각해 질 것으로 예상된다.

소거연산을 전체 플래시 메모리 영역에 골고루 분산시켜 저장 장치의 수명을 연장시키기 위해 다양한 마모평준화(wear-leveling)기법이 연구되어 왔다[3]. 하지만 기존 기법들은 복잡한 쓰레기 수집(garbage collection) 정책에 의존하거나 모든 블록의 소거횟수를 유지 관리하기 위해 많은 비용을 지불해야 하는 문제가 있었다.

본 논문에서는 마모 제어 영역(Regulation Pool)을 활용한 마모평준화 기법을 제안한다. 제안하는 기법은 FTL(Flash Translation Layer)의 메타데이터 저장영역을 Regulation Pool로 활용하여 소거횟수가 낮은 블록에 쓰기 및 소거연산을 집중시키는 단순한 방식으로 Pool 내부의 소거횟수를 평준화한다. Regulation Pool에 속한 블록들은 다른 영역의 블록들과 교체되어 전체 플래시 메모리 블록의 마모평준화에 기여한다.

본 논문의 구성은 다음과 같다. 2장에서는 FTL에 관한 기본적인 내용과 기존 연구된 마모평준화 기법들을 소개한다. 3장에서는 제안하는 기법의 적용 대상인 RS -FTL에 대해 간략히 소개한다. 4장에서는 Regulation Pool을 활용한 마모평준화 기법의 기본 원리 및 동작 과정에 관해 기술하며, 5장에서는 RS-FTL에 본 기법을 적용하였을 때의 마모평준화 효율 향상에 관한 실험 결과를 제시한다. 마지막으로 6장에서는 결론 및 향후 과제를 정리한다.

2. 기존 연구

2.1 FTL

플래시 메모리는 쓰기 동작이 이루어지기 전 소거동작이 선행되어야 한다. 또한 읽기 및 쓰기 동작은 페이지 단위로 수행되지만 소거 동작은 블록 단위로 수행되는 비대칭성이 존재하며, 단위 블록 당 소거 횟수가 제한되는 등 기존의 저장 매체와는 다른 특성을 가진다. FTL은 플래시 메모리의 고유한 특성을 고려하여 상위 계층 소프트웨어에게 섹터 단위의 읽기 및 쓰기 요청을 제공하는 소프트웨어 계층이다. FTL은 저장장치의 성능과 안정성을 향상시키기 위해 다양한 관리기법을 사용하며, 그 중 대표적인 것으로는 주소 사상기법, 배드 블록 관리 기법, 그리고 마모평준화 기법이 있다.

주소 사상 기법은 호스트 시스템의 논리주소를 플래시 메모리상의 물리주소로 사상시키는 기법으로서, 사상 단위에 따라 페이지사상[4,5] 및 블록사상[6-9]으로 나누어진다. 주소 사상기법에 의해 FTL은 내부적으로 섹터 단위의 읽기 및 쓰기 동작 요청들을 플래시 메모리에서의 페이지 단위 읽기 및 쓰기 요청과 블록 단위의 소거 요청으로 변환한다. 플래시 메모리는 제자리 갱신이 되지 않기 때문에 저장장치가 사용됨에 따라 무효화된 데이터 공간이 필연적으로 발생하며, 이를 재활용하기 위한 블록 병합작업 또는 쓰레기 수집 동작이 수행되어야 한다.

배드 블록은 정상적인 읽기/쓰기/소거 동작이 수행되지 않는 블록을 의미하며 플래시 메모리 쓰기/소거 동작 중에 임의적으로 발생할 수 있다. 배드 블록 관리 기법은 이러한 블록들을 식별하여 사전에 준비된 정상 블록과 교체하는 동작을 수행한다.

마지막으로 FTL은 플래시 메모리 전체 영역을 균일하게 사용하여 플래시 메모리 기반 저장장치의 수명을 연장하기 위해 마모평준화 기법을 적용한다. 2.2절에서 기존 마모평준화 기법에 대해 간략하게 설명하겠다.

2.2 마모평준화 기법

기존 마모평준화 기법은 쓰레기 수집 동작의 희생블록 선정 정책을 변경하는 방식[10,11], 전체 블록간 소거 횟수 차이를 계속 검사하며 조건에 의해 블록의 내용을 교환하는 방식[12], 소거 동작이 일정 기간 이상 수행되지 않은 블록에 대하여 쓰레기 수집 동작을 수행하는 방식[13]등이 있다.

첫 번째 방식은 쓰레기 수집 동작의 희생블록 선정 시 블록 내 유효한 페이지의 개수(블록 활용도)[4] 뿐 아니라 블록의 소거 횟수도 함께 고려하여 마모평준화를 수행한다. 위 방식은 하나의 기법으로 쓰레기 수집 효율 향상 및 마모평준화의 효과를 동시에 추구하므로 복잡하고 많은 계산 자원을 소모하는 문제가 있다.

두 번째 방식은 쓰레기 수집 동작과 독립적으로 전체 공간에 대한 블록 소거횟수의 편차를 계속 감시하며, 이 수치가 지정된 값을 넘어서는 순간 마모평준화를 위한 작업이 수행된다. 이때 소거횟수가 가장 높은 블록과 가장 낮은 블록의 내용 및 사상정보를 교환하는데, 이는 참조확률이 높은(hot) 데이터일수록 블록의 소거횟수를 빠르게 증가시킨다는 가정에 기초한다. 위 방식은 데이터의 참조확률이 계속 변화하는 경우 정확한 동작을 보장하기 어렵다.

마지막 방식은 전체 블록 중 참조확률이 낮은 데이터가 저장되어 있는 블록을 효과적으로 구분할 수 있는 방안을 제시한다. 이를 통해 판별된 블록들은 쓰레기 수집 동작의 희생블록으로 선정되고, 참조확률이 낮은 데이터가 점유하고 있던 블록에 hot 데이터가 저장되어 블록간의 소거횟수가 균일하게 유지될 것을 기대한다. 위 방식은 hot 데이터에 대한 구분 방식이 명확하지 않다는 단점이 있다.

또한 앞서 기술된 방식들은 공통적으로 플래시 메모리의 전체 블록에 대한 소거횟수를 별도의 자료구조로 관리해야 하기 때문에 메모리 사용량이 증가하고, 이를 관리하기 위한 추가적인 비용 부담이 필요하다.

3. RS(Region Shift)-FTL

RS-FTL은 센서 네트워크 환경과 같이 자원 사용량이 매우 제한된 환경에서 최소한의 메모리를 사용하면서도 효과적으로 동작하는 FTL의 구현을 목표로 개발되었다[14]. RS-FTL은 사상 정보 관리에 필요한 메모리 사용량을 최소화하기 위해 파일 시스템에서 접근하는 논리주소와 데이터영역에 대한 논리주소를 동일하게 사상하는 고정 주소 사상 기법을 사용하며, 마모평준화를 위해 영역이동 기법을 이용한다. 그림 1은 RS-FTL의 영역 구분을 나타내며, 각 영역의 기능은 다음과 같다.

- Data 영역: 사용자로부터 전달되는 데이터를 보관한다. 파일 시스템 관점에서는 이 영역의 크기가 실질적인 저장 장치의 크기로 보이게 된다.
- Metadata Logging 영역: 동작 중 예기치 않은 전원 중단 등에 의해 발생할 수 있는 FTL 상태 불일치 문제를 해결하기 위한 공간이다. 쓰기 및 소거 동작, 블록 병합, 영역이동 등과 같은 FTL 상태 변화에 관련된 주요 동작이 수행되기 전, 복원에 필요한 상태 정보를 미리 정의된 형태로 이 영역에 순차적으로 기록한다.

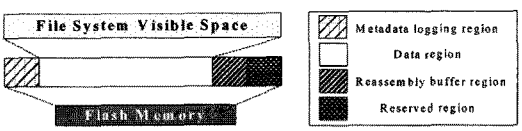


그림 1 RS-FTL의 영역 구분

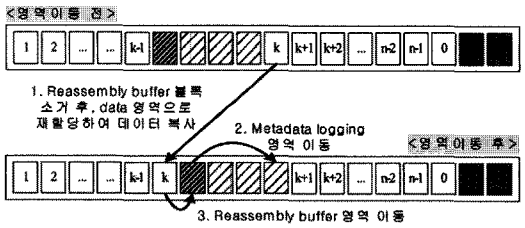


그림 2 RS-FTL의 영역이동 기법

- Reassembly Buffer 영역: Data 영역에 대한 쓰기 동작 수행 시 블록 단위의 병합 과정을 수행하기 위한 임시 버퍼의 역할을 수행한다. 또한 성능 향상을 위한 쓰기 버퍼로도 이용된다.
- Reserved 영역: 동작 중 발생할 수 있는 배드 블록을 처리하기 위한 영역이다. 일정 수의 정상 블록들을 유지하면서, 다른 영역에서 배드 블록이 발생하는 경우 이들과 교체하여 준다.

그림 2는 RS-FTL의 영역이동 기법의 동작을 나타낸다. RS-FTL은 마모평준화를 위해 모든 블록의 소거횟수를 유지하는 대신, 전체 플래시 메모리 영역에 가해진 총 소거횟수만을 유지한다. 이 값이 미리 지정된 값을 넘어가면 각 영역의 경계에 위치하고 있는 블록들을 하나씩 서로 교체하는 영역이동을 수행한다.

영역이동에 의해 플래시 메모리에 속한 모든 블록들은 workload의 특성이 다른 영역들을 한번씩 경험할 수 있다. 이를 통해 영역별 특성의 차이가 상쇄되어 최종적으로 모든 블록들의 소거횟수 역시 균일하게 유지되는 것을 기대할 수 있다. 이때 영역이동에 의한 마모평준화의 효과는 미리 정의된 영역이동 주기(총 소거횟수/단위시간)의 값에 영향을 받는다. 영역이동 주기가 짧을수록 영역이동을 자주 수행하기 때문에 전체 블록 간 소거횟수는 고르게 유지할 수 있지만, 영역이동을 위한 블록 소거 및 유효한 데이터 복사의 비용은 많이 지불하게 된다. 반대로 영역이동 주기가 길게 설정된 경우 마모평준화의 효과는 줄어들지만 영역이동을 위한 오버헤드는 줄어드는 trade-off 관계를 보이게 된다.

RS-FTL의 영역이동 기법은 2.2절에서 소개된 기존 마모평준화 기법들에 비해 메모리 사용 및 관리부담이 적다. 하지만 마모평준화를 주기적인 영역이동에만 의존하기 때문에, 사용자 workload가 일부 영역에 편중되거나 동적으로 변화하는 경우에 대해서는 효과적으로 대응하기 어렵다.

4. Regulation Pool을 활용한 마모평준화 기법

본 논문에서 제안하는 기법은 FTL이 관리하는 플래시 메모리 블록들이 관리 목적에 따라 서로 다른 용도로 사용될 수 있다는 점에 착안한다. 3절에서 살펴본 RS-FTL은 전체 플래시 메모리 공간을 4개의 영역으로 구분하고 있으며, 각 영역은 용도에 따라 각기 다른 접근 양상을 보인다. Data 영역의 경우 쓰기 및 소거 동작이 수행되는 블록은 사용자의 workload에 의해 결정되기 때문에 FTL은 블록의 참조패턴을 예측하기 어렵다. 반면 Metadata Logging 영역의 경우 쓰기 및 소거 동작의 대상 블록은 FTL이 결정한다. 즉 해당 영역에 속한 블록들에 대해 workload를 균일하게 분산시킬 수도 있지만, 경우에 따라서는 특정 블록에 집중시키는 방법을 통해 영역 내의 블록 소거횟수를 자유롭게 조절하는 것이 가능하다.

본 논문에서는 메타데이터 저장 공간과 같이 FTL이 workload 패턴을 자유롭게 변경할 수 있는 영역을 Regulation Pool이라고 정의한다. FTL은 Regulation Pool에 속한 각 블록들의 소거횟수를 메모리에 보관하며, 소거횟수가 낮은 블록에 대해 연산을 집중하고 반대로 소거횟수가 높은 블록에 대해서는 연산을 방지하는 방법을 통해 블록들의 소거횟수를 균일하게 유지할 수 있다. 또한 Regulation Pool에 속한 블록 간에 마모 평준화가 이루어지면, 이들을 다른 영역에 속한 블록들과 교체하여 플래시 메모리 전체 블록에 대한 마모평준화 역시 유도할 수 있다.

Regulation Pool 내부에서 소거횟수의 평준화를 이루는 방식은 블록에 소속된 데이터의 참조확률을 예측하는 것이 아닌, 데이터의 참조확률을 직접 조정하는 방식이기 때문에 그 효과가 명확하다. 또한 다른 마모 평준화 기법에서 요구되는 복잡한 쓰레기 수거 정책의 적용에 비해 계산 부담이 적다. 마지막으로 블록의 소거횟수를 관리하기 위한 메모리 공간 역시 절약할 수 있다. 메타데이터 저장 공간과 같이 Regulation Pool로 지정이 가능한 영역의 크기는 전체 플래시 메모리 크기에 비해 일반적으로 매우 작기 때문이다.

5장에서는 본 기법을 RS-FTL에 적용하여 마모평준화의 효율성을 확인해 본다.

5. 기법 평가

5.1 Regulation Pool 기법의 RS-FTL 적용

기법 적용을 위해 RS-FTL의 Metadata Logging 영역을 Regulation Pool로 지정하고, logging 블록 선정 방식을 기존 round-robin 방식에서 그림 3과 같이 수정하였다. RS-FTL은 영역이동 기법을 사용하고 있기 때문에 Regulation Pool에 속한 블록과 다른 영역에 속한 블록의 교체 과정은 자연스럽게 이루어진다.

그림 3에서 소거횟수가 가장 낮은 블록을 대상블록으로 선정하기에 앞서 Regulation Pool에 속한 블록의 소

```
* Notations
- MLB : array of metadata logging blocks
- c_idx : index of current metadata logging block
- n_mlb : mumber of metadata logging blocks
- EC(A) : erase count of block A
- ECavg : average block erase count

metadata_logging ()
1: if ( MLB[c_idx] has no free pages ) then
2:    c_idx = get_next_logging_idx ( c_idx );
3:    erase MLB[c_idx]
4: write log into MLB[c_idx]

get_next_logging_idx ( int c_idx )
1: for i = 0 to n_mlb-1 do
2:    if( i ≠ c_idx and EC( MLB[i] ) < ECavg ) then
3:       return i;
4: return (index of the least erased metadata logging block except c_idx)
```

그림 3 Metadata Logging 영역 내부 동작 의사 코드

거 횟수를 평균값과 비교하는 이유는, RS-FTL의 영역 이동 기법의 특성상 하나의 블록이 Regulation Pool을 빠져나가기 직전까지 최대한 소거횟수 평준화의 효과를 얻기 위함이다.

5.2 실험 구성

실험은 NAND 플래시 시뮬레이터를 기반으로 진행되었으며, RS-FTL에 다양한 workload를 인가하며 Regulation Pool 기법 적용 유무에 따른 마모평준화 효율을 비교하였다. 평가 기준은 플래시 메모리에 속한 전체 블록 중 가장 많이 지워진 블록과 가장 적게 지워진 블록 간 소거횟수의 차이로 삼았다.

실험에서 사용된 NAND 플래시 메모리의 구성 및 RS-FTL의 영역 설정은 표 1과 같다. 본 실험에서는 Regulation Pool 적용에 의한 마모평준화의 효율 향상만을 비교하기 위해 시뮬레이션 수행 중 배드 블록은 발생하지 않는다고 가정하고, 페이지 크기 및 블록당 페이지 개수는 임의로 설정하였다. Metadata Logging 영역의 블록 개수는 동작에 필요한 최소한의 개수인 3개로 설정하였다.

시뮬레이션의 workload는 synthetic workload를 사용하였으며, 각 workload의 특성은 다음과 같다.

- Uniform workload: 각 주소 영역에 대한 참조확률이 모두 동일하게 분포한다.
- Hot-Cold workload: 각 주소 영역에 대한 참조확률이 영역 간 비대칭성을 이룬다. 본 실험에서는 20%의 주소 공간에 80%의 참조가 집중되며, 시뮬레이션이 진행되는 동안 정해진 주소 영역 별 참조확률은 변하지 않는다.
- LRU workload: 각 주소 영역에 대한 참조확률이 참조의 최신성에 비례하여 높아진다. 따라서 주소 영역 별 참조확률은 시뮬레이션이 진행되는 동안 시간에 따라 변화하는 특성을 보인다.

표 1 NAND 플래시 메모리 및 RS-FTL 구성

구분	내용
페이지 크기	2KB
블록당 페이지 개수	64개
블록 개수	2048개
Reassembly Buffer 영역 블록 개수	4개
Reserved 영역 블록 개수	0개
Metadata Logging 영역 블록 개수	3개

5.3 실험 결과

그림 4는 각 workload가 진행되는 동안 플래시 메모리 전체 블록에 대한 소거횟수 편차의 변화 양상을 그래프로 나타낸 것이다. 왼쪽의 그래프는 Regulation Pool 기법을 적용하지 않았을 때의, 오른쪽의 그래프는 기법을 적용하였을 때의 마모평준화 효율을 보여주고 있다. 해당 값이 작을수록 전체 플래시 메모리 블록간 소거횟수가 균일하게 분포하고 있다는 것을 의미한다.

블록 소거횟수의 편차를 샘플링 하는 주기는 각 workload 상에서 영역이동 주기가 블록의 총 개수의 배수만큼 발생하는 시점으로 일치시켰다. 이는 RS-FTL 고유의 특성을 반영하기 위함으로, 모든 블록이 영역이동을 한번씩 경험한 시점을 나타낸다.

본 실험에서 RS-FTL의 마모평준화 효율에 영향을 미치는 인자는 workload, 영역이동 주기(RS_period), 그리고 Regulation Pool 기법 적용 유무 세 가지이다. 그림 4의 좌/우 그래프를 비교하여 보았을 때, 기법 적

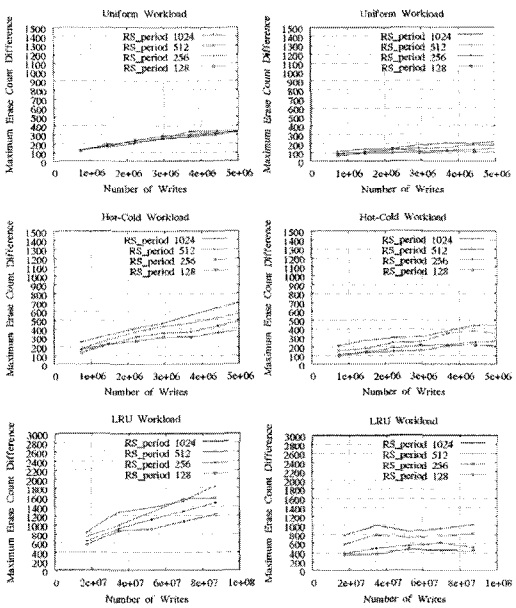
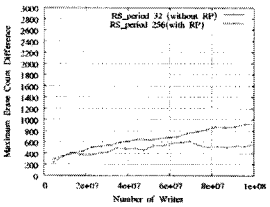


그림 4 마모평준화 성능 평가
(좌: 기법 미적용, 우: 기법 적용)



	Total erase count
RS_period 32 (without RP)	12,690,601
RS_period 256 (with RP)	12,198,620

그림 5 기법 적용에 의한 동작 오버헤드 감소

용에 의해 workload 및 영역이동 주기 설정 변화에 상관없이 블록 소거횟수의 편차가 약 40% 정도 감소하는 것을 확인할 수 있다. 또한 쓰기 동작의 진행에 따른 블록 소거횟수 편차의 증가 추세 역시 감소하는 것을 확인할 수 있다.

그림 5는 LRU workload에 대하여 Regulation Pool 기법을 적용하지 않고 영역이동 주기를 32로 설정한 실험과 기법을 적용하고 영역이동 주기를 256으로 설정한 실험 결과를 비교하고 있다. Workload가 동일할 경우 전체 플래시 메모리 블록에 가해진 소거동작의 총 횟수는 영역이동 주기 설정에 따라 달라지는 것을 예상할 수 있다. 이 차이를 마모평준화를 위한 오버헤드로 보았을 때, 그림에서 보이는 바와 같이 블록 간 소거횟수의 편차는 영역이동 주기를 256으로 설정하고 Regulation Pool 기법을 적용한 경우가 더 적지만 전체 영역에 가해진 소거횟수는 오히려 약 50만회 정도 적다. 즉, RS-FTL에서 Regulation Pool 기법 적용으로 보다 높은 수준의 마모평준화 효과를 보다 낮은 오버헤드를 통해 달성할 수 있음을 확인할 수 있다.

6. 결론 및 향후 과제

본 논문에서는 Regulation Pool을 활용한 플래시 메모리의 마모평준화 기법을 제시하였다. 본 기법은 FTL이 관리하는 플래시 메모리 블록들이 관리 목적에 따라 서로 다른 용도로 사용되고 있다는 점에 착안하여 단순한 연산 및 적은 자원 사용으로 마모평준화를 수행할 수 있는 방안을 제시하였다. 제안하는 기법을 RS-FTL에 적용한 결과, 전체 플래시 메모리의 블록 간 소거횟수의 편차가 약 40% 정도 줄어드는 것을 확인할 수 있었다.

본 논문에서 제안하는 기법이 다른 FTL에서도 적용되기 위해서는 RS-FTL의 경우와 같이 Regulation Pool에 소속된 블록을 다른 영역에 속한 블록들과 교체하는 과정이 추가되어야 한다. 이를 위해 각 FTL의 동작 특성을 고려한 효율적인 블록 교체 방식이 구현될 필요가 있다. 향후 다양한 FTL에 본 기법을 적용하여 기법의 효용성을 검증하는 연구가 진행될 예정이다.

참 고 문 헌

[1] S. K. Lee, S. L. Min, Y. K. Cho, "Current trends on flash memory technology," *Journal of KIISE*, vol.24, no.12, pp.99-106, Dec. 2006. (in Korean)

[2] J.-K. Kim, "Improving system performance and longevity with a new NAND flash architecture," Flash Memory Summit 2009.

[3] E. Gal, S. Toledo, "Algorithms and data structures for flash memories," *ACM Computing Surveys*, vol.37, no.2, pp.138-163, Jun. 2005.

[4] A. Kawaguchi, S. Nishioka, H. Motoda, "A flash-memory based file system," *Proc. of the USENIX 1995 Technical Conference*, 1995.

[5] M. Wu, W. Zwaenpoel, "eNVy: A non-volatile, main memory storage system," *Proc. of the 6th International Conference on Architectural Support for Programming Languages and Operating Systems*, vol.29, no.11, pp.86-97, Nov. 1994.

[6] J. Kim, J. M. Kim, S. H. Noh, S. L. Min, Y. Cho, "A space-efficient flash translation layer for CompactFlash systems," *IEEE Transactions on Consumer Electronics*, vol.48, no.2, pp.366-375, May. 2002.

[7] A. Ban, R. Hasharon, "Flash file system optimized for page-mode flash technologies," *United States Patent*, no.5,637,425, Aug. 1999.

[8] J.-U. Kang, H. Jo, J.-S. Kim, J. Lee, "A super-block-based flash translation layer for NAND flash memory," *Proc. of the 6th ACM & IEEE International Conference on Embedded Software*, pp.161-170, 2006.

[9] S.-W. Lee, D.-J. Park, T.-S. Chung, D.-H. Lee, S. Park, H.-J. Song, "A log buffer-based flash translation layer using fully-associative sector translation," *ACM Transactions on Embedded Computing Systems (TECS)*, vol.6, no.3, Jul. 2007.

[10] M.-L. Chiang, P. C. H. Lee, R.-C. Chang, "Managing flash memory in personal communication devices," *Proc. of the 1997 IEEE International Symposium on Consumer Electronics(ISCE'97)*, pp.177-182, 1997.

[11] H.-J. Kim, S.-G. Lee, "A new flash memory management for flash storage system," *Proc. of the 23rd International Computer Software and Applications Conference*, pp.284-289, 1999.

[12] L.-P. Chang, "On efficient wear-leveling for large-scale flash-memory storage systems," *Proc. of the 2007 ACM symposium on Applied computing*, pp.1126-1130, 2007.

[13] Y.-H. Chang, J.-W. Hsieh, T.-W. Kuo, "Improving flash wear-leveling by proactively moving static data," *IEEE Transactions on Computers*, vol.59, no.1, pp.53-65, Jan. 2010.

[14] Y. H. Bae, "A flash memory file system for mobile devices," Doctorial Thesis, School of Computer Engineering, Seoul National University, 2006.