다중-채널 및 다중-웨이 반도체 디스크를 위한 플래시 변환 계층

(Flash Translation Layer for the Multi-channel and Multi-way Solid State Disk)

박 현 철 † 신 동 군 ††
(Hyunchul Park) (Dongkun Shin)

요 약 플래시 메모리는 전력 소비가 적고 처리속도가 빨라 임베디드 시스템의 저장 매체로서 많은 연구가 이루어져왔다. 특히, 최근에는 플래시메모리로 구성된 반도체 디스크(Solid state disk, SSD)가 하드디스크를 점점 대체하고 있는 추세이다. 현재 SSD는 성능을 높이기 위해서 병렬성을 이용한 다중채널과 다중웨이를 사용하고 있다. 이 구조에서는 연속된 여러 개의 블록들로 구성된 슈퍼블록단위로 플래시 메모리에 기록하게 된다. 본 논문은 병렬처리를 최적화하기 위해 SSD의 버퍼를 비울 때 희생 슈퍼블록을 선정하고 재구성하는 방법을 제안하고 있다. 실험을 통해서 희생 슈퍼블록 선정 방법을 바꾸는 것으로 슈퍼블록단위의 쓰기 횟수를 35% 줄일 수 있고, 슈퍼블록 구성 방법을 달리하여 9%를 추가적으로 더 줄일 수 있었다.

키워드 : 플래시 메모리, SSD, FTL, 다중 채널

Abstract Flash memory has several features such as low-power consumption and fast access so that there has been various research on using flash memory as new storage. Especially the Solid State Disk which is composed of flash memory chips has recently replaced the hard disk. At present, SSD adopts the multi-channel and multi-way architecture to exploit advantages of parallel access. In this architecture, data are written on SSD in a unit of a superblock which is composed of multiple blocks in which some blocks are put together. This paper proposes two schemes of selecting, segmenting and re-composing victim superblocks to optimize concurrent processing when a buffer flush occurs. The experimental results show that 35% of superblock- based write operations is reduced by selecting victims and additional 9% by composition of superblock.

Key words : Flash Memory, SSD, FTL, Multi-channel

· 이 논문은 2008년 정부(교육과학기술부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임(KRF-2008-314-D00351)
· 이 논문은 제34회 추계학술대회에서 '다중채널 반도체 디스크를 위한 플래시 변환 계층'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 성균관대학교 정보통신공학부
hcpark@skku.edu
†† 정 회 원 : 성균관대학교 정보통신공학부 교수
dongkun@skku.edu
(Corresponding author임)

논문접수 : 2008년 12월 19일
심사완료 : 2009년 7월 1일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제15권 제9호(2009.9)

1. 서 론

최근 다방면에서 플래시 메모리가 활용되는 비중은 급속도로 커졌다. 이는 빠른 처리 속도, 비휘발성, 무소음, 저전력, 내구성 같은 특징 때문이다. 특히 여러 개의 낸드(NAND) 플래시 칩과 제어장치로 구성된 SSD[1]는 차세대 저장 장치로써 주목받고 있다.

그러나 플래시 메모리는 덮어쓰기가 불가능하고, 읽기와 쓰기는 페이지 단위인데 지우기는 연속한 페이지들의 집합인 블록 단위로 수행되는 단점이 있다. 따라서 기존의 파일시스템에게 플래시 메모리가 하드디스크인 것처럼 에뮬레이션(Emulation)해주는 플래시 변환 계층(Flash Translation Layer, FTL)[2,3]이 필요하다.

현재 SSD는 다수의 플래시 메모리 칩들을 병렬로 배치하는 다중채널(Multi-channel) 다중웨이(Multi-way)구조를 활용하고 있다[4]. 이 구조에서는 다수의 플래시 메모리 칩들에 동시접근이 가능하다. 이를 뒷받침하기 위해 슈퍼블록이라는 맵핑 단위를 도입하였고, 슈퍼블록 단위에 특화된 FTL의 고안이 필요하다. 본 논문에서는 SSD의 병렬성을 최대한 활용하기 위한 SSD의 쓰기 버퍼 관리 방법과 슈퍼블록 구성기법을 제시하고 있다.

본 논문의 내용은 다음과 같다. 2장에서는 다중채널 다중웨이 SSD 구조의 특성과 문제점을 설명하고, 3장에서는 SSD의 병렬성을 활용하는 방법을 제시한다. 4장에서는 시뮬레이션을 통한 실험 결과를 보여주며, 5장에서는 결론을 맺고, 추가 연구 주제를 소개한다.

2. 다중웨이 다중채널 SSD

본 논문에서 다루고 있는 SSD의 구조는 그림 1과 같이 4 채널 & 2 웨이 SSD이다. 기존의 SATA, SCSI등의 입출력 인터페이스의 대역폭은 150MB/s인데 비해, 낸드 플래시 버스의 대역폭은 40MB/s에 불과하다. 그러나 각 버스의 낸드 제어기(Controller)들을 통하여 4개의 버스에 동시에 데이터를 전송한다면 전체 대역폭이 160MB/s가 되어서 기존의 I/O 인터페이스를 충분히

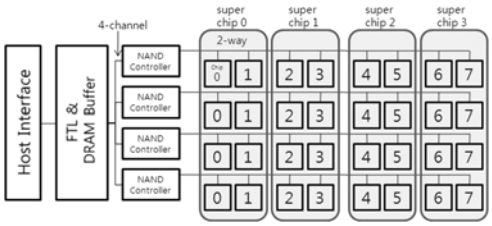

그림 1 4-channel & 2-way SSD 구조

지원할 수 있다. 또한 버스 내에서도 8개의 칩들이 4쌍으로 묶여서 한쌍의 칩들에 대해서 동시에 접근이 가능하다. 따라서 총 8개의 칩에서 동시에 명령을 처리할 수 있기 때문에 플래시 메모리 칩의 쓰기 속도의 최대 8배의 속도로 쓸 수 있으며, 4개의 버스에서 동시에 데이터를 전송받을 수 있으므로 플래시 메모리 칩의 원래 읽기 속도의 최대 4배의 속도로 읽을 수 있다. 그리고 SSD 내부에는 DRAM 버퍼가 있어서 호스트로부터의 쓰기 요청을 버퍼링(Buffering)한다.

2.1 슈퍼블록 단위의 명령 수행

본 구조의 장점을 활용하기 위해서는 쓰기 버퍼에서 데이터를 모아서 각각의 버스와 칩들에 동시에 보내줄 수 있어야 한다. 이를 위해 8개의 블록으로 구성된 슈퍼블록(Superblock) 단위로 전송하면 효율적이다.

슈퍼블록의 구성방법을 알아보면, 우선 각 버스에서 동일한 위치에 있는 4쌍의 칩들을 모아서 슈퍼칩(Superchip)으로 간주한다. 즉 4개의 버스의 0번 칩들과 1번 칩들을 모아서 0번 슈퍼칩으로 간주한다. 그리고 슈퍼칩의 8개의 칩들에서 같은 번호의 블록 8개를 모은 슈퍼블록을 작업 수행의 기본 단위로 사용한다. 그림 2는 이 구성을 나타낸다. 읽을 때에는 한 개의 페이지만 읽을 수 있고, 플래시 메모리 칩 수준의 명령은 페이지와 블록단위로 수행된다.

그림 2 슈퍼블록의 구성

2.2 슈퍼블록 전송 시 문제점

제시된 구조의 쓰기 버퍼에서 슈퍼블록단위로 데이터를 모을 때, 일부 칩들에 쓸 데이터가 없는 불완전한 슈퍼블록이 발생할 수 있다. 이 경우, 맵핑(Mapping)단위를 유지하기 위해서 갱신되지 않는 블록까지 복사하는 낭비가 발생하게 된다.

그림 3은 쓰기 버퍼의 내용을 플래시 메모리에 기록할 때 논리 슈퍼블록(Logical Superblock, LSB) 0번부터 순서대로 비울 경우의 상황을 나타낸다. 쓰기 버퍼를 비우기 전 LSB들이 같은 번호의 물리적 슈퍼블록(Physical Superblock, PSB)에 맵핑되어 있다고 가정하자. 쓰기 버퍼의 사용량이 일정 수준을 넘어서 비워야하는 경우, LSB 0에 갱신할 내용이 있기 때문에 빈 PSB 4를 할당받아서 기존의 내용과 새로운 내용을 합쳐서 PSB 4에 쓴다. 그리고 PSB 0은 무효화(Invalidate)되면서 LSB 0이 PSB 4에 맵핑된다. 그러나 블록 0, 1, 4에 해당하는 데이터만 갱신되어서 PSB 4로 이동되고, 블록 2, 3, 5, 6, 7의 데이터는 PSB 4로 그냥 이동되는 것이다. 만약, 이후에 갱신되지 않았던 곳이 갱신되면 새로운 PSB를 할당받아서 써야하므로 비효율적이다.

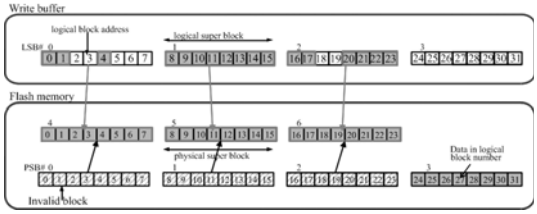

그림 3 슈퍼블록 단위의 쓰기의 문제점

3. 다중채널 다중웨이 SSD 구조를 위한 FTL

불완전한 슈퍼블록의 쓰기를 막기 위해 이 논문에서는 두 가지 방법을 제시하고 있다. 첫째는 버퍼를 비울때 내보낼 희생 슈퍼블록을 결정해서 일부만 보내는 것이다. 둘째는 슈퍼블록에서 갱신할 데이터를 다른 슈퍼블록에서 가지고 와서 모든 칩에 갱신할 데이터를 보낼 수 있는 완전한 슈퍼블록을 구성하고, 다중 수준 주소 맵핑(Multi-level address mapping)으로 맵핑하는 것이다.

3.1 희생 슈퍼블록 결정

DRAM 버퍼는 덮어쓰기가 가능하므로 버퍼 적중률(Hit rate)를 높이고, 데이터를 최대한 모아서 보내는 것이 쓰기 횟수를 줄일 수 있다. 따라서 데이터가 가장 많이 모인 LSB를 희생 슈퍼블록으로 선정할 수 있다. 그러나 상대적으로 크기가 작은 데이터가 계속 남게 되어, 버퍼의 공간을 차지하고 버퍼 적중률을 떨어뜨릴 수 있다. 그러므로 접근된 지 오래된 것일수록 희생 대상으로 선택하는 LRU(Least Recently Used)기법을 사용해야 한다. 하지만 LRU만 따르다보면 접근된 지 오래되고 데이터가 조금 모인 불완전한 LSB가 선정될 수 있다. 따라서 희생 슈퍼블록을 선정할 때 LSB에 모인 데이터의 양과 LRU를 동시에 고려하여 선택해야 한다. 이를 위해 본 논문에서는 다음과 같은 희생 슈퍼블록 선정 휴리스틱(Heuristics)을 제안한다.

$$R(i) = \log_n (X_i + 1) + \frac{Y_i}{n} \quad (1)$$

위의 식 (1)에서 $R(i)$는 LSB i의 우선순위를 의미한다. n은 쓰기 버퍼에 있는 LSB들의 개수이며, X_i는 LSB i의 LRU 정도(0:가장 최근에 접근, n-1:가장 접근한지 오래됨), Y_i는 LSB i에 데이터가 모여진 양의 순위(0: 가장 크기가 작음, n-1: 가장 크기가 큼)이다. 위의 식을 통하여 우선순위가 가장 큰 것을 희생 슈퍼블록으로 선택한다.

3.2 다중수준 블록 맵핑(Multi-level Block Mapping)

희생 슈퍼블록들이 불완전하다면 이들을 모아서 완전한 슈퍼블록으로 만들어 쓰는 것이 성능향상에 유리하다. 이를 위해서 슈퍼블록보다 작은 단위의 맵핑 테이블이 필요하다.

본 논문에서는 슈퍼블록들을 조립하는 두 가지 방법을 제안하는데 블록들의 물리적 위치를 고려하지 않고 최대한 조합하는 Best-fit Composition(이하 Best-fit)과 블록들의 물리적 위치를 고려한 Location-fit Composition(이하 Location-fit)이다.

그림 4는 Best-fit에서 LSB들을 이용하여 가상 슈퍼블록(Virtual Superblock, VSB)을 만들어서 플래시 메모리에 쓰는 과정을 보여준다. 이 때, 각 블록들의 LSB 내에서의 위치와 상관없이 VSB를 구성하게 된다. 그림 4에서는 4개의 LSB를 가지고 3개의 VSB를 만드는 예를 보여주고 있다. 예를 들어 VSB 0은 LSB 0, 2, 3에서 연속하는 한 쌍의 블록들을 모은 것이고, 새로운 내용이 없는 페이지나 블록은 플래시 메모리에 기록하게 될 때 원래 내용을 복사해온다. 이렇게 완성된 3개의 VSB만을 플래시 메모리에 기록하면 되므로 기존의 방법보다 훨씬 플래시 메모리에 대한 I/O 요청량을 줄일 수 있게 된다. 본 논문에서는 조립의 최소 단위는 LSB 내에서 연속한 2개의 블록으로 했다. 따라서 VSB 0을 구성할 때 새로운 내용이 없는 5번, 24번 블록은 플래시 메모리 칩에서 복사해야 한다.

VSB의 조립을 위해서 슈퍼블록보다 작은 단위의 맵핑 테이블이 필요하기 때문에 그림 5처럼 L1 테이블과 L2 테이블을 이용한다. L1 테이블은 블록 4개 단위의 맵핑 정보를 기록하며, PSB의 번호와 PSB 내에서의 위치를 가리키는 1bit를 포함하고 있다. L2 테이블은 블록 2개 단위의 맵핑 정보를 기록하고 있으며, PSB 번호와 PSB 내에서의 위치를 알려주는 2bits를 포함하고 있다. L0 테이블은 기본적으로 PSB 번호를 가리키지만 조각나게 되면 조각난 부분이 L1, L2 테이블의 위치를 가리키고, 현재 해당 LSB가 얼마나 조각나 있는지 알리는 3bits를 포함하고 있다.

물론, SSD의 DRAM의 가용 공간이 L1테이블과 L2 테이블을 위한 추가 공간으로 인하여 줄어들게 된다. 그렇지만 32GB SSD에서 다중 수준 블록 맵핑을 위한 맵핑 테이블 크기는 80.25KB로 슈퍼블록 맵핑 시 60KB가 필요한 것에 비하면 추가 공간의 크기는 크다고 할 수 없고, 슈퍼페이지 맵핑 시 필요한 5.25MB의 2%도 안되는 크기이다. 32GB SLC SSD에서 각 맵핑 방법의 테이블 크기는 다음과 같다. 여기서 L1테이블과 L2테이블은 각각 2048개의 엔트리를 가질 수 있다.

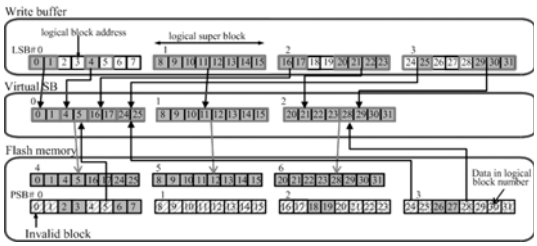

그림 4 Best-fit Superblock Composition

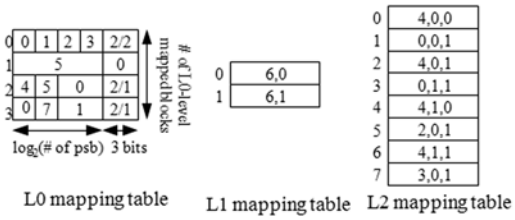

그림 5 Best-fit Composition 맵핑 테이블

- 슈퍼블록의 크기 : 1MB, 슈퍼블록의 개수 : 2^{15}개
- 슈퍼페이지의 크기 : 16KB, 슈퍼페이지의 개수 : 2^{21}개
- 슈퍼페이지 맵핑 테이블 크기 : 21bit * 2^{21} = 5.25MB
- 슈퍼블록 맵핑 테이블 크기 : 15bit * 2^{15} = 60KB
- 다중수준 블록 맵핑 :
  - L0테이블 : (16bit + 3bit) * 215 = 72KB
  - L1테이블 : (15bit + 1bit) * 211 = 4KB
  - L2테이블 : (15bit + 2bit) * 211 = 4.25KB
  - 총 크기 : 80.25KB

LSB 2번에서 왼쪽 두 쌍의 블록은 L2 테이블의 4번, 5번 엔트리에 PSB 정보가 기록되어 있으며, 오른쪽 두 쌍의 블록은 L1 테이블의 0번 엔트리에 PSB 정보가 기록되어 있다. 만약 메모리상에 테이블을 위한 공간이 더 이상 없을 경우 보조 테이블들이 가리키는 블록들을 새로운 PSB에 옮겨서 L1테이블과 L2테이블의 내용을 L0 테이블로 합쳐야하는 추가 작업이 요구된다.

Best-fit 방법은 SSD의 용량을 최대한 활용할 수 있지만, 블록들의 위치가 다른 칩으로 바뀌게 되어 일단 데이터를 DRAM으로 읽어왔다가 쓰는 횟수가 많아진다. 이를 막기 위해 플래시 메모리가 지원하는 copy-back 명령이 있다. 이 명령은 같은 칩 내부에서 데이터가 이동하는 경우 내부 페이지 버퍼를 거치게 한다. 그

러므로 가급적 블록들의 위치를 유지한 채 VSB을 구성하면 후에 갱신이 일어나도 copy-back만으로 새로운 데이터를 쓸 수 있게 된다. 다른 칩에 써야할 경우, 기존 방법처럼 읽어온 후 쓰기를 수행한다. 이것이 Location-fit방법이며 그림 6에 표현되어있다.

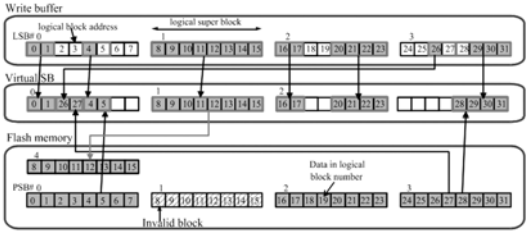

그림 6 Location-fit Superblock Composition

위의 그림 6과 같이 VSB를 구성하는 데 있어서 블록들의 원래 위치를 지키는 것을 볼 수 있다. 예를 들어 VSB 0을 구성하기 위하여 LSB 0, 3이 참여한다. VSB 0의 마지막 2개 블록에 LSB 2의 22, 23번 블록이 들어갈 수도 있다. 그러나 Best-fit과 Location-fit에서는 남는 공간을 채우기 위해 연속된 4개의 블록이나 8개의 블록을 억지로 분리시키지 않는다. 따라서 VSB 0은 버퍼에 남게 되고, 대신 완전한 VSB 2가 희생 대상으로 선택되는 것이다. 이 방법에선 L1, L2테이블에서 위치 정보를 포함할 필요가 없다. 하지만 위치를 지키다보니 완전한 슈퍼블록을 구성하기 힘들어지는 경우가 발생하게 되며, 버퍼의 용량 부족 등의 원인으로 인하여 어쩔 수 없이 써야할 경우 플래시 메모리 내에서 데이터의 불필요한 이동이 발생하게 된다.

4. 실험과 분석

4.1 실험 조건

실험은 16MB DRAM 버퍼와 32GB의 4 채널, 2 웨이 SSD의 시뮬레이터를 만들어서 수행하였다. 맵핑의 수준이 최소 블록 2개 단위로 이루어지기 때문에 블록 수준의 명령을 처리하도록 하였다.

실험의 낸드 플래시 메모리는 SLC(Single Layer Cell)이다. 그리고 한 페이지는 2KB이고, 한 블록은 64개의 페이지로 이루어져있다. 한 개의 낸드 플래시 메모리 칩은 8192개의 블록으로 이루어져 1GB이 된다. 쓰기 버퍼의 데이터들은 슈퍼블록단위로 관리한다. 사용량이 90%를 넘으면 버퍼를 비우기 시작하고, 가용량이 전체 용량의 50% 이상이 될 때까지 수행한다. 맵핑 테이블들의 크기에 제한을 두지 않고 DRAM에서 맵핑 테이블들이 차지하고 있는 영역은 고려하지 않았다.

폐영역 회수(Garbage Collection)방법과 웨어레벨링(Wear-leveling)방법은 이 논문의 주요 관심사가 아니므로 할당받을 블록이 없을 때에만 무효화된 블록을 지우고, 블록을 할당받을 때 블록의 지워진 횟수는 고려하지 않고 지워진 횟수가 십만 번이 넘으면 더 이상 지울 수 없는 블록으로 관리하게 하였다.

실험에 쓰인 트레이스(Trace)는 데스크탑 컴퓨터에서 프로그램 설치 및 실행, MP3 플레이어, 웹브라우저와 게임 등의 다양한 응용프로그램을 실행하며 수집한 트레이스와 2GB의 JPEG 파일을 복사하면서 생기는 순차적인 접근들을 수집한 트레이스를 사용하였다.

4.2 실험결과

그림 7은 데스크탑 컴퓨터에서 다양한 작업을 수행할 때 발생한 쓰기 요청을 보여주고 있다. 세로축은 슈퍼블록단위의 쓰기 횟수를 나타내며 가로축은 기법들을 나타내고 있다.

여기서 Default는 슈퍼블록 단위를 그대로 사용하는 방법이다. Size는 슈퍼블록에 모인 데이터의 양을 기준으로 선정하는 것이고, LRU기법과 Size를 같이 고려하는 경우가 LRU+Size이다. 가장 우측의 FlushAll로 표기된 부분은 DRAM 버퍼를 비울 때 모든 데이터를 내보내는 가장 기본적인 방법이다. 그리고 막대의 형태는 완전한 슈퍼블록과 불완전한 슈퍼블록을 나타낸다.

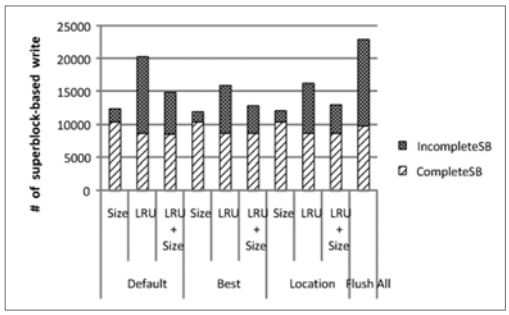

그림 7 데스크톱에서 다양한 작업 수행 시 성능 비교

가장 기본적인 방법인 FlushAll을 보면 불완전한 슈퍼블록이 많이 발생하고 있음을 확인할 수 있다. 희생 대상 선정 방법들을 비교하기 위해 Default를 보면, Size가 슈퍼블록 쓰기 횟수를 가장 많이 줄이고 불완전한 슈퍼블록도 가장 적게 발생시키는 것을 볼 수 있다. 그러나 특정 데이터가 버퍼에 계속 머무르는 것을 막기 위해 LRU를 도입했지만 LRU는 불완전한 슈퍼블록들로 인해 성능이 떨어진다.

따라서 LRU+Size를 사용하면 데이터가 골고루 희생 대상으로 선정되면서 성능 개선도 이룰 수 있다. 이 때 Size는 FlushAll에 비하여 46%만큼 슈퍼블록 쓰기 횟수를 줄였고 LRU+Size도 35%의 감소를 보여준다.

LRU나 LRU+Size의 경우 불완전한 슈퍼블록이 많이 발생하기 때문에 Best-fit과 Location-fit을 통해서 쓰기 횟수를 더 줄일 수 있다. Default와 Size의 조합에서는 불완전한 슈퍼블록의 비중이 작기 때문에 Best-fit으로 인한 개선을 기대하기 힘들고, Default와 LRU+Size의 조합에선 Best-fit 기법을 통해 추가적으로 9%를 줄일 수 있음을 알 수 있다. Location-fit 기법의 경우 Best-fit 기법보다 완전한 슈퍼블록을 만드는 데 효율이 약간 떨어졌다.

그림 8은 2GB의 JPEG 파일을 복사할 때 SSD의 플래시 메모리에 보내진 쓰기 요청을 보여주고 있다. 그래프의 형식은 그림 7과 동일하다. 이 경우, 순차적인 접근이 주로 발생하므로 FlushAll에서도 완전한 슈퍼블록이 많이 생긴다. 또한 Default에서 Size와의 조합보다 LRU와의 조합이 슈퍼블록의 쓰기 횟수가 더 줄어들었다. 그 이유는 Size에선 동일한 슈퍼블록에 연속적으로 쓰기 명령이 들어오는 도중에 그 슈퍼블록이 희생대상으로 선정되어 써지기 때문이다. 그러나 LRU 경우엔 가장 최근에 접근된 슈퍼블록이므로 희생대상으로 선정되지 않는다. LRU+Size는 접근된 지 오래되면서 병렬 처리의 효율을 높일 수 있는 슈퍼블록을 선택하므로 쓰기 횟수를 더 줄일 수 있다. 여기에 Best-fit을 적용하여 최종적으로 FlushAll에 비해 22%만큼 쓰기 횟수를 줄일 수 있다.

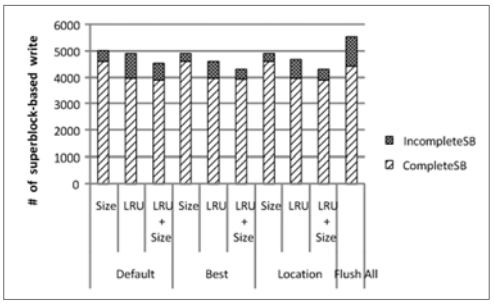

그림 8 2GB의 JPEG파일을 복사할 때 성능 비교

그림 9에서는 그림 7과 동일한 실험에서 Best-fit과 Location-fit을 비교하기 위한 결과를 보여주고 있다. 가로축은 실험한 방법들을 나타내고 있으며 세로축은 블록단위의 쓰기 횟수를 나타낸다. R&W(read-and-write)는 copy-back 대신 우선 DRAM으로 읽어왔다가 다른 칩의 블록에 쓰는 것을 의미한다.

R&W를 수행해야하는 경우는, 첫 번째, 슈퍼블록 내에서 블록의 위치가 바뀌어서 다른 버스의 칩으로 데이터를 옮겨야 하는 경우이고, 두 번째는 다른 슈퍼칩으로 데이터를 옮겨야 하는 경우이다. copy-back은 수행 중에 다른 칩에 명령을 보낼 수 있지만, R&W는 읽고 쓰면서 버스를 점유하기 때문에 다른 칩에 명령을 보낼 수 없고 시간이 더 걸린다는 단점이 있다.

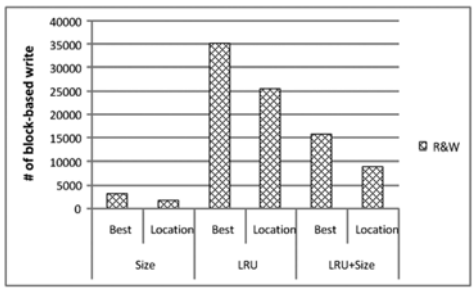

그림 9 Best-fit과 Location-fit의 비교

블록들의 위치를 고려하지 않는 Best-fit과 달리, Location-fit 기법은 블록들의 위치를 지키기 위해 노력하므로 R&W의 횟수가 줄어들고, 처리 속도가 더 빨라진다. 하지만 Best-fit 기법은 플래시 메모리의 공간을 최대한 사용할 수 있기 때문에 각각의 장단점이 있다.

5. 결 론

본 논문에선 다중채널 다중웨이 SSD가 슈퍼블록 맵핑을 사용하면 병렬성을 제대로 활용하지 못할 수 있음을 보였다. 그리고 병렬 처리의 효율을 극대화하기 위하여 버퍼에서 플래시 메모리로 내보낼 희생 슈퍼블록을 선정하는 방법을 제시하였고, 슈퍼블록을 구성할 수 있는 다양한 방법들과 이를 지원하기 위한 다중레벨 맵핑 방법도 제시하였다. 또한 버퍼를 비우는 방법과 슈퍼블록을 구성하는 방법에 따라서 44%까지 슈퍼블록 쓰기 횟수를 줄일 수 있다는 것을 실험을 통해서 보였다. 마지막으로 L1 테이블과 L2 테이블을 효율적으로 관리하는 방법을 연구하는 것이 추가 연구 주제이다.

참 고 문 헌

[1] Samsung Electronics. "Solid State Drive Data Sheet, "http://www.samsung.com/global/business/semiconductor/products/flash/Products_FlashSSD.html.

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

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

[4] J.U.Kang, J.S.Kim, C.Park, H.Park, J.LEE, "A Multi-channel Architecture for High-performance NAND flash-based storage system," *Journal of Systems Architecture*, vol.53, no.9, pp.644-658, 2007.