반도체및디스플레이장비학회지 제7권 제3호(2008년 9월)
Journal of the Semiconductor & Display Equipment Technology, Vol. 7, No. 3. September 2008.

다중 플래시 메모리 기반 파일시스템의 성능개선을 위한 파일시스템

박 제 호†

†단국대학교 컴퓨터과학과

File System for Performance Improvement in Multiple Flash Memory Chips

Je Ho Park†

†Computer Science, Dankook University

ABSTRACT

Application of flash memory in mobile and ubiquitous related devices is rapidly being increased due to its low price and high performance. In addition, some notebook computers currently come out into market with a SSD(Solid State Disk) instead of hard-drive based storage system. Regarding this trend, applications need to increase the storage capacity using multiple flash memory chips for larger capacity sooner or later. Flash memory based storage subsystem should resolve the performance bottleneck for writing in perspective of speed and lifetime according to its physical property. In order to make flash memory storage work with tangible performance, reclaiming of invalid regions needs to be controlled in a particular manner to decrease the number of erasures and to distribute the erasures uniformly over the whole memory space as much as possible. In this paper, we study the performance of flash memory recycling algorithms and demonstrate that the proposed algorithm shows acceptable performance for flash memory storage with multiple chips. The proposed cleaning method partitions the memory space into candidate memory regions, to be reclaimed as free, by utilizing threshold values. The proposed algorithm handles the storage system in multi-layered style. The impact of the proposed policies is evaluated through a number of experiments.

Key Words : Flash Memory File System, Block Cleaning, Even Wear-Leveling, SSD

1. 서 론

근래 플래시 메모리 칩은 용량 면에서32Gbyte 를 저장할 수 있는 기술을 확보하여 차세대 저장매체로서 실제적인 시장의 크기는 날로 커지고 있다.

이러한 저장용량의 증가는 휴대폰, MP3플레이어, PMP(Portalble Multimedia Player), 네비게이션, 디지털 카메라 등의 모바일 기기의 보조 저장매체의 용량 증가에 대한 요구를 만족시키려는 노력의 결과이다. 또한, 유비쿼터스 분야에서의 플래시 메모리를 저장매체로 사용하는 비중도 늘고 있으며, 일부 노트북은 하드 디스크 기반 저장매체 대신 플래시 메모리 기반 SSD(Solid State Disk)를 저장매체로 채택하고 있다[4,6,7,9,10]. 이러한 경향은 플래시 메모리 기반 저장장치가 가지는 부피, 외부충격에 대한 내구성, 소비전력, 응답시간이라는 측면에서의 우수성에 기인한다. 이러한 플래시 메모리 기반 저장매체를 사용하는 소프트웨어는 플래시 메모리 드라이버, 플래시 변환 계층(FTL, Flash Translation Layer), 플래시 메모리 파일 시스템 등으로 구성되는 다계층 구조를 가지고 있다[9].

일반적으로 플래시 메모리 기반 시스템을 지원하는 FTL은 플래시 메모리의 특성 중 저장매체로 사용하는데 장애가 되는 균등 소거 알고리즘을 포함하고 있다. 플래시 메모리는 자체적 갱신 동작을 지원하지 않는다. 또한, 저장된 자료의 소거를 위해서는 읽기/쓰기를 할 수 있는 영역단위를 포함하는 미리 지정된 영역(블록)을 일시에 소거해야 한다. 균등 소거는 특정 부분에 소거가 집중되어 발생되는 플래시 메모리의 장애를 막기 위하여 사용된다. 이 논문에서는 단일 칩에서 사용되는 균등 소거 알고리즘들이 다수 플래시 메모리로 구성되

†E-mail : dk_jhpark@dankook.ac.kr

는 저장매체에서 사용될 때의 영향을 분석하고 이로 인해 성능 저하를 방지할 수 있는 알고리즘을 제안한다. 또한, 성능 분석을 위해 수행한 실험의 결과를 예시한다.

2. 플래시 메모리 개요

Fig. 1에서 보는 바와 같이, 플래시 메모리는 생산자에 의해 정의되는 블록들로 구성된다. 하나의 블록은 읽기/쓰기가 가능한 특정 수의 페이지들로 구성된다. 플래시 메모리의 기본 동작 및 특성과 필요 소요시간은 Table 1에 보여지고 있다. 복사 동작은 특정 페이지를 다른 블록에 있는 페이지로 복사할 때 사용되며, 대부분의 NAND 플래시 메모리 제품에서 지원되지만, 일부 MLC NAND 플래시 메모리에서는 지원되지 않는다. 읽기 및 쓰기 동작의 소요시간은 RAM과 플래시 메모리 사이의 데이터 전송시간이 포함되어 있지 않다[1].

플래시 메모리의 특성 상 상대적으로 느린 쓰기 및 복사의 소요시간, 제한된 갱신 횟수라는 물리적 특성은 제한적 상황을 야기시킬 수 있다. 또한, 플래시 메모리의 소거는 블록 단위로만 가능하기 때문에 기존의 메모리에서 사용하는 방법과는 다른 갱신(upate)과정이 필요하다. In-place 갱신은 갱신 대상 페이지가 속하는 블록에 있는 유효한 다른 페이지들을 시스템 버퍼에 임시로 저장 한 뒤, 해당 블록을 소거한 후, 버퍼에 저장된 페이지들과 새로 갱신되는 페이지를 함께 기존의 블록에 쓰기 동작을 통해 저장을 한다.

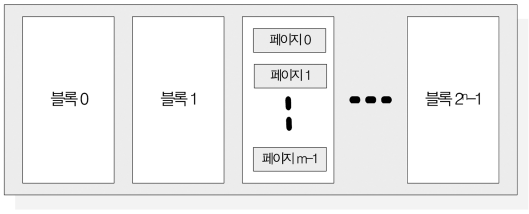

Fig. 1. Configuration of Flash Memory.

Table 1. Characteristics of Flash Memory

동작	동작단위	특성값
읽기	페이지	25 microsec
쓰기	페이지	200~300 microsec
복사	페이지	200~300 microsec
소거	블록	2 millisec
크기	블록	64/128 Kbytes
소거한도	블록	100 K ~ 1,000 K

우리가 고려해야 하는 또 다른 플래시 메모리의 특성은 블록 갱신 횟수에 대한 제한이다. 만일 한 블록의 갱신 횟수가 제한된 범위에 도달하면 해당 블록은 손상되고, 해당 영역에 대한 쓰기 시도는 장애가 발생시켜 적절한 반응 속도를 보장할 수가 없게 된다. 이를 방지하기 위해서는 발생되는 갱신들을 전체 플래시 메모리 공간에 균등하게 분포되도록 조절하는 것이 절대적으로 필요하다. Non-in place방법론은 갱신 과정에 포함된 읽기와 쓰기 동작을 분리하여 수행한다[8].

메모리 재활용을 위한 플래시 메모리 공간을 확보하기 위해서는, 부분적으로 무효화된 자료가 저장된 페이지가 속하는 블록에서 유효한 자료(페이지)들을 다른 블록으로 이동시킨 뒤, 해당 블록을 소거한다. 소거 대상 블록의 선택은 클리닝(cleaning) 정책에 의해 결정된다. 블록 소거는 소요 시간이 클 뿐 아니라 전력 소모량이 다른 동작에 비해 많기 때문에, 클리닝 정책의 목표는 소거 횟수를 최소화하여 전력 소비와 반응 속도 측면에서 시스템 성능을 개선하는 동시에 일부 블록에 소거가 집중되는 것을 방지하여 균등 소거(even wear leveling)를 이루는 것이다.

3. 소거 대상 선택 방법론

3.1. 기존의 방법론

가장 원초적인 소거 대상 블록 선택 방법론은 무작위적 방법이다. 이외에 기존의 방법론들은 최적(최소/최대) 블록 속성값을 전체 메모리 공간에서 탐색한다. 탐욕적 방법론은 대상 블록들 중에서(무효 블록 수+미사용 블록 수) 값의 최대값을 가지는 블록을 소거한다. 비용/편익 기간 방법론은 특정 블록에 대해 마지막 페이지 무효화 시간과 계산 수행까지의 시간을 age라 하고, 그 블록의 유효 자료 비율을 u라고 정의하고(age∗(1-u))/(2∗u)의 최대값을 가지는 블록을 소거한다[3]. 이러한 전역검색 기반 방법론은 플래시 메모리의 대용량화에 따라 탐색 영역이 커져 검색에 따는 비용의 증가가 불가피하다. 따라서, 시스템 운영에 필요한 비용의 급증은 플래시 메모리에 대한 기대 성능에 부응하지 못하는 결과를 가져 올 수 있다.

Fig. 2는 다수의 플래시 메모리 모듈을 이용하여 저장매체를 구성할 경우의 일반적 모델이다. 구성하고자 하는 메모리의 용량이 하나의 칩으로 구현되지 않을 경우, 여러 개의 칩은 버스 구조에 의해 전체 메모리 공간을 구성하기 위해 연결된다.

한 개의 플래시 메모리 모듈에 포함된 블록의 개수를 ChipBNo이라고 하자. 또한 적어도 한 개 이상 유

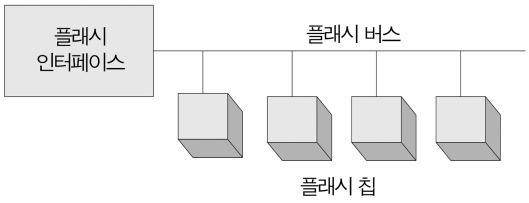

Fig. 2. Multiple Flash Memory based Storage.

효 데이터 블록을 포함하는 블록들의 전체 메모리에 대한 비율을 ChipUtil 이라고 정의하자. 시스템에 전역 탐색을 적용할 때, 최적값 탐색에 필요한 블록의 수 ChipSpace 는 다음과 정의된다.

$$ChipSpace = [ChipBNo * ChipUtil] \quad (1)$$

따라서, n개의 플래시 메모리 모듈로 구성되는 시스템 전체에 대한 검색 영역 SysSpace는 다음과 같이 정의된다.

$$SysSpace = \sum_{i=0}^{n-1} [ChipBNo * ChipUtil_i] \quad (2)$$

플래시 메모리 기반 저장매체가 하나 이상의 플래시 메모리 모듈로 구성 될 경우, 검색에 필요한 비용 증가는 최종 저장매체의 성능을 크게 저하시킬 것이다. 이러한 결과는 특정 플래시 메모리 칩에 속하는 페이지를 위하여 소거하는 블록을 다른 칩에서 찾을 경우 발생되는 이동 비용이 커질수록 심화된다. 하지만, 전체 시스템의 균등 소거를 위해서는 칩 사이의 페이지 이동을 완전히 배제할 수도 없다.

본 논문에서 제안하는 방법론은 소거 대상 검색 영역을 축소하여 기존의 방법론의 검색 비용을 감소시키고, 동시에 비최적값 기반 방법론이 보이는 성능의 감소를 최소화하는 방법론을 제안하고자 한다. 제안하는 방법론은 소거 대상 블록을 결정할 때, 단일 메모리 모듈 내부에서의 수행되어야 하는 소거와 해당 모듈 외부에서 수행되어야 하는 소거를 경험적 방법론에 의해 결정을 하게 된다. 칩 계층에서 소거 대상 칩을 선택하는 방법과 칩 내부 블록들 중에서 소거 대상 블록을 선택하는 알고리즘을 일반화하여 칩 도메인(domain)과 블록 도메인에 적용되는 속성값만을 변화시켜 체계의 단순화와 성능 개선을 위해 제안하는 알고리즘은 임계값을 이용한다.

3.2. 임계값 기반 소거 대상 선택

임계값 기반 소거 대상 선택은 탐색 영역을 다수의 동일한 영역 크기를 가지는 하부 탐색 영역으로의 분할한다. 최적분할을 필요한 카디널리티(Cardinality) K 값은 실험적인 방법론에 결정한다. 각 계층별 하부 탐색 영역의 이상적인 크기 SubSize는 다음과 같이 정의된다.

$$SubSize = \left\lceil \frac{SearchSpac\ e}{K} \right\rceil \quad (3)$$

따라서, 소거 대상을 계층적으로 선택할 때, 탐색 대상은 식 (3)에 의해 한정되고, 각 하부 영역은 탐색 우선 순위를 나타내는 대표 속성값을 가지게 된다. 분할 {Region(K), ⋯ ,Region(1)}에 속하는 소거 대상들은 분할의 최저한계치과 최대한계치 안에 속하는 값을 가진다. 전역 검색에서 사용하는 속성값과 분할 시 사용되는 속성값이 같다고 가정할 때, 최상위 대표 속성값에 상응하는 하부 영역을 탐색한 결과는 적용된 알고리즘에 따라 차이를 보일 수 있으며, 크게 무작위 선택과 전역 탐색 기반 선택을 사용할 수 있다.

플래시 메모리 모듈들 전체에 걸쳐 전역 탐색을 하여 선택된 최적의 소거 대상 블록 B_{opt}는 계층적 방법에 의해 메모리 모듈을 대상으로 하여 선택을 한 다음 선택된 모듈 내부에서 선택된 B_K와는 차이를 보일 것으로 예측된다. 이는 계층적 선택 방법론으로서는 피할 수 없는 단점으로 보이지만, 메모리 모듈 상태에서 전체 결과를 고려를 하기 때문에 전체적인 결과는 상이할 것으로 예상 할 수 있다.

여기서 우리가 고려를 하여야 하는 것은 하부영역에서 소거 대상을 어떻게 선택할 것인가 하는 문제이다. 적절한 속성값을 가진 블록의 소거는 균등성을 향상시켜 결과적으로 레벨링을 개선한다는 점을 고려하여 전체 영역에 무작위 선택을 적용하는 것보다 하부 탐색 영역 Region(K)에 무작위 선택을 적용한 경우가 보다 개선된 레벨링 효과를 가져올 것이다. 하부영역 Region(K)에 속하는 블록들은 높은 최적성에 따라 분할되었으므로, Region(K)에 무작위 방법을 적용하는 것이 전체 소거 대상 집합에서 최적성의 고려 없이 무작위 선택을 적용한 것보다 레벨링 기여도 면에서 우수할 것이다.

전역 탐색은 소거 대상의 선택을 최적성에 따라 결정하기 때문에 동일한 속성값 함수 AttrVal을 적용할 때, 전역 탐색 정책과 분할 기반 정책은 성능 측면에서 차이는 자연스런 현상이다. 하지만, 전역 탐색의 경우 비교에 의한 선택이 필수적이므로 탐색 비용은 분할 기반 정책보다 높다. 식 (2)에서 보이는 모든 블록을 대상으로 하는 방법과 메모리 모듈의 선택과 모듈 내부의 분할에 의한 탐색은 비용 면에서 많은 차이는 명백하다.

3.3. 임계값 기반 분할

제안하는 방법론은 탐욕적 전체검색에서 사용하는 속성값을 K개의 하부 영역으로 분할한다. 따라서, 속성값 G는 한 소거 대상에 존재하는 무효 페이지 수와 데이터 페이지로 사용되지 않은 페이지 수의 합으로 정의한다. B를 한 소거 대상에 속하는 블록수로 정의할 때, G는 다음의 영역을 갖는다.

$$1 \le G \le (B-1) \tag{4}$$

이 영역을 K 개 하부 영역으로 분할하기 위해 구간 간격 W를 다음과 같이 정의한다.

$$W = \left\lceil \frac{B-2}{K} \right\rceil \tag{5}$$

각 하부 영역은 대표 속성값 P를 가진다고 가정하면, 대표 속성값은 1부터 K까지의 값을 가진다. 따라서, 대표 속성값 P를 가지는 하부 영역은 다음과 같이 하한 임계값 L와 상한 임계값 U를 가지도록 정의한다.

$$L = 1 + (P-1)*W \tag{6}$$

$$U = L + (W-1) \tag{7}$$

하부 영역 대표 속성값이 K 인 경우는 상한 임계값을 (B – 1)로 정의한다. 본 논문에서 제안하는 플래시 메모리 관리의 유효성과 자유 세그먼트 선택에 따른 영향성은 시뮬레이션 기반 실험을 통해 검증하였다.

4. 구현 사례 및 결과 분석

4.1. 구현사례

실험의 목표는 첫째로 제안하는 플래시 메모리 관리의 유효성을 실험을 통해 검증하는데 있다. 실험에 사용된 플래시 메모리 모델은 전형적인 모델을 따라 구성하였으며, 전체 플래시 메모리 기반 저장매체의 모델링은 일반적 모델을 사용하여 구현하였다[3,10]. 실험은 시스템에 접근하는 논리 페이지를 번호를 태스크 발생기를 사용하여 생성했으며, 생성된 접근 논리 번호는 플래시 파일시스템에 의해 물리적 페이지 번호로 변환되어, 해당 페이지에 읽기/쓰기를 수행시켰다.

4.2. 성능 분석을 위한 실험

본 논문에서는 시스템의 크기를 32 GByte로, 블록 크기는 일반적인 구성을 따라 64 KByte로 설정하고, 페이지 크기는 2 KByte로 설정하였다. 시스템 활용도 역시 성능에 크게 영향을 미치는 것으로 알려져 있다[3,7]. 성능 면에서 관찰할 때 성능 측면에서 병목현상이 나타나는 시스템 활용도 80%를 채택하였다. 실험은 시스템 구성을 위해 사용된 플래시 메모리 모듈의 개수를 증가시키면서 수행되었다.

접근 집약성은 태스크 발생기에 사용 빈도수가 높은 영역에 대한 접근(Hot Access)과 사용 빈도수가 낮은 영역에 대한 접근(Cold Access)을 설정하고, 접근 발생 시 접근의 종류를 조절하여 접근 집약성을 구현하였으며, Hot Access 영역에 대한 백분율을 80%로 설정하였다. 성능 분석을 위해 구현된 알고리즘들은 기존의 무작위 선택 탐욕적 선택(ChipGreedy), 비용/편익 기반 선택 (ChipCostBenefit) 방법론들과 제안하는 임계값 기반(ChipMultiset) 방법론들이다. 플래시 메모리 모듈을 선택하는 방법과 메모리 모듈 내부에서 소거 대상 블록을 선택하는 방법은 같은 알고리즘을 사용하였으며, 선택에 사용되는 속성값은 앞에서 설명한 것과 같이 메모리 모듈 선택과 블록 선택에 서로 다른 속성값을 사용한다. 임계값 기반 방법론에 사용된 카디널리티는 메모리 모듈 내부에 대한 기존의 성능 분석을 통해 결정된 값을 사용하였다.

4.3. 성능 분석을 위한 실험

Table 2는 블록 소거 당 지원 접근수를 예시하고 있으며, 범례 OneMultiset은 메모리 모듈의 구분이 없이 블록 전체 영역에 임계값 기반 방법론을 적용하였을 때의 성능을 보여 준다. 실제 수치로는 전체 블록 공간에 임계값 기반 방법론을 적용하였을 때 27.86, 계층별로 따로 적용을 하였을 때 26.04를 보이고 있다. 다른 방법론도 모두 이 구간에서 성능을 보이고 있어, 실제로 필요한 블록 소거의 횟수는 크게 차이가 나지 않는 것으로 보인다.

Table 3은 단일 페이지 갱신을 위해 필요한 블록 소거가 필요할 때, 페이지가 속하는 메모리 모듈 외부에서 소거 대상 블록을 선택하는 경우를 전체 소거에 대한 백분율을 보여준다. 결과치에서 보는 바와 같이 같은 크기의 메모리 공간에 구성 메모리 모듈의 개수가 늘어날 수로 외부 모듈에서 소거가 발생하는 확률은 커지며, 방법론에 따라 큰 차이를 보이지 않고 있다.

Table 2. Access per Block Erasure

Legend\Modules	2	4	8	16
OneMultiset	27.86	27.86	27.70	27.86
ChipGreedy	27.93	27.40	27.40	27.55
ChipCostBenefit	27.78	27.47	27.70	27.40
ChipMultiset	27.47	27.10	26.74	26.04

Table 3. Percentage of Inter-chip Erasure

Legend\Modules	2	4	8	16
OneMultiset	52	74	85	93
ChipGreedy	52	79	89	93
ChipCostBenefit	49	76	86	94
ChipMultiset	49	73	86	92

Table 4. Even Wear Leveling

Legend\Modules	2	4	8	16
OneMultiset	0.0027	0.0027	0.0028	0.0027
ChipGreedy	0.0027	0.0028	0.0028	0.0028
ChipCostBenefit	0.0027	0.0028	0.0028	0.0028
ChipMultiset	0.0026	0.0028	0.0029	0.0029

Table 4는 전체 메모리 시스템의 균등 소거 레벨을 보이고 있으며, 그림에서는 큰 차이로 보이지만, 실제 값은 메모리 모듈의 개수가 16개일 때, 최저 0.0027, 최고 0.0029로 차이가 거의 없다고 할 수 있다. 여기서 유의를 해야 할 점은 실제 비슷한 성능을 얻는데 소요된 비용이라는 점이다. 위에서 설명한 것과 같이 임계값 기반 방법론은 전역 검색 기반 방법론과 비교를 하였을 때, 최소의 비용을 사용하면서도 비슷한 성능을 보이고 있는 것이다.

Fig. 3는 균등 소거 레벨의 표준편차를 보이고 있는데, 비용/편익 기반 선택 방법론과 임계값 기반 분할 방법론은 거의 비슷한 우수한 성능을 보이고 있다.

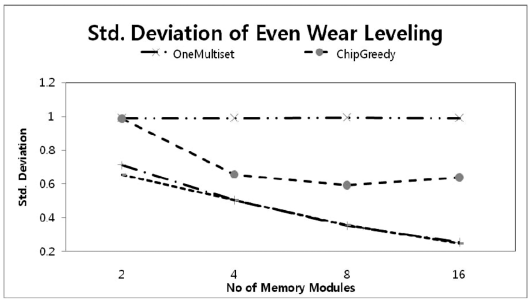

Fig. 3. Standard Deviation of Even Wear Leveling.

4. 결 론

제안된 방법론은 선택에 필요한 비용을 최대 한 줄이고, 성능의 저하를 최소화하자는 목표를 가지고 개발되었다. 현재 플래시 메모리의 응용이 단일 메모리 모듈을 이용하는 응용에서 다수의 모듈을 사용하는 시스템으로 발전하고 있어, 성능 구현에 필요한 비용을 최소화하여 실제 구현에서 플래시 메모리가 가지는 장점을 최대화하고자 하는 노력이다. 실험결과에서 보는 바와 같이 제안하는 방법론은 수용하는 메모리 모듈의 개수가 늘어날수록 장점을 보이고 있으며, 차후 필요한 비용을 더욱 감소시키기 위해, 메모리 모듈 선택 시 필요한 경우를 판단할 수 있다면, 향후 제안된 방법론의 성능은 더욱 증대될 것이다.

감사의 글

본 연구는 단국대학교 2008학년도 교내연구비 지원으로 이루어졌습니다.

참고문헌

1. NAND Flash Memory SmartMedia Data Book, Samsung Electronics, 2006.
2. Mei-Ling Chiang, Paul C.H. Lee and Ruei-Chuan Chang. “Using Data Clustering to Improve Cleaning Performance for Flash Memory.” Software-Practice and Experience, 29(3) : 267-290, 1999.
3. Mei-Ling Chiang and Ruei-Chuan Chang. “Cleaning Policies in Mobile Computers Using Flash Memory.” Journal of Systems and Software, 48(3) : 213-231, 1999.
4. Joshua B. Fryman et al. “Energy-efficient Network Memory for Ubiquitous Devices.”, IEEE Micro, 23(5) : 60-70, 2003.
5. Jen-Wei Hsieh, Li-Pin Chang and Tei-Wei Kuo. “Efficient On-line Identification of Hot Data for Flash-Memory Management”, In SAC, pp. 838-842, 2005.
6. Brian Marsh, Fred Douglis and P. Krishman. “Flash Memory File Caching for Mobile Computers.” In HICSS(1), pp. 451-461, 1994.
7. Sang Lyul Min and Eyee Hyun Nam. “Current Trends in Flash Memory Technology” In Proceedings of ASP-DAC 2006, Yokohama, Japan, January 24-27, 2006.
8. M. Rosenblum and J. K. Ousterhout. “The Design and Implementation of a Log-Structured File System.” ACM Trans. Computer Systems, 10(1) : 26-52,1992.
9. 곽종철, 민상렬, 박장석, “Flash Memory 기반 임베디드 소프트웨어 기술 개발,” 정보과학회지 제25권 제6호, pp. 5-10, 2007.
10. 배영현, “고성능 플래시 메모리 SSD(Solid State Disk) 설계기술,” 정보과학회지 제25권 제6호, pp. 5-10, 2007.

접수일자: 2008년8월1일, 심사일자: 2008년 8월20일
게재확정일자: 2008년 8월22일