DRAM/MRAM 하이브리드 메인 메모리와 플래시메모리 저장 장치를 고려한 버퍼 캐시 기법*

양수현*, 류연승
*명지대학교 컴퓨터공학과
e-mail : soohyun121@gmail.com, ysryu@mju.ac.kr

A Buffer Cache Scheme Considering both DRAM/MRAM Hybrid Main Memory and Flash Memory Storages

Soo-Hyun Yang*, Yeon-Seung Ryu
*Dept. of Computer Engineering, Myong-Ji University

요 약

모바일 환경에서 전력 손실이 중요한 문제 중 하나가 됨에 따라, MRAM 과 플래시메모리와 같은 비 휘발성 메모리가 차세대 모바일 컴퓨터에 널리 사용될 것이다. 본 논문에서는 DRAM/MRAM 하이브리드 메인 메모리의 제한적인 쓰기 연산 성능을 고려한 효율적인 버퍼 캐시 기법을 연구했다. 제안한 기법은 MRAM 의 제한적인 쓰기 연산 성능을 고려하고 플래시 메모리 저장 장치의 삭제 연산 횟수를 최소화한다.

1. 서론

대부분의 최신 운영체제(OS)는 속도가 느린 보조 저장 장치에 의해 제한되는 I/O 성능을 향상시키기 위해서 일반적으로 버퍼 캐시 메커니즘을 사용한다. 지난 수십 년 동안, 버퍼 캐시 기법은 DRAM 기반의 메인 메모리와 하드디스크 기반의 보조 스토리지로 구현되었다.

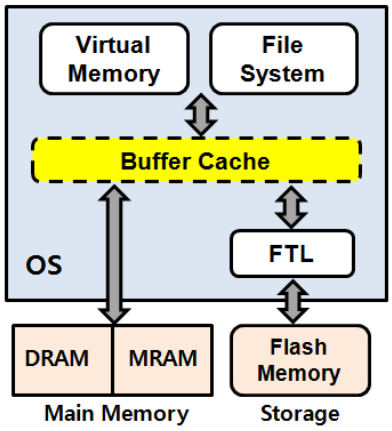
그러나 최근 연구에서 DRAM 기반의 메인 메모리는 전체 시스템 전력의 상당 부분을 소비하는 것으로 나타났다. 이러한 문제는 배터리로 작동되는 스마트폰과 태블릿 PC 와 같은 모바일 컴퓨터에서 심각한 문제를 야기한다. 다행스럽게도, PRAM(Phase change RAM)과 MRAM(Magnetic RAM)과 같은 저전력 비 휘발성 메모리가 개발되었다. 비 휘발성 메모리 중에서도 MRAM 은 높은 밀도와 저 전력 소비로 모바일 정보기기, 각종 컴퓨터에 사용될 차세대 메모리로 각광 받고 있다. DRAM 기반의 메인 메모리의 에너지 손실을 해결하기 위해서, 최근 연구에서 MRAM 기반 메인 메모리 기법을 소개했다[4]. MRAM 은 전기도체의 저항이 주변 자기장에 따라 변화하는 자기저항효과(magneto resistance effect)를 이용해 정보를 저장하는 방식으로, 단일 트랜지스터 위에 자기터널접합 구조를 갖는 MTJ(magnetic tunneling junction) 셀로써 구성된다. MRAM 의 읽기 성능(지연 시간,전력 소비)은 DRAM 과 비교될만하지만, 쓰기 성능(지연 시간,전력 소비)는 DRAM 보다 떨어진다(지연 시간 1.25-2 배,전력 소비 5-10 배). 그러나 MRAM 은 DRAM, 플래시 메모리 등 기존 메모리에 비해 재생 속도가 빠르며, 소비전력이 적고, 비 휘발성이다. 또한, 셀 크기가 작아 집적도가 높은 장점을 가지고 있다.

대부분의 모바일 컴퓨터에서, 하드 디스크보다 플래시 메모리가 적은 전력을 소비하고 빠르기 때문에, 일반적으로 NAND 플래시 메모리 기반의 저장 장치가 사용된다. 그러나, 플래시 메모리는 저장되어 있는 데이터를 바로 변경할 수 없으며 데이터가 저장 되어 있는 블록에 대해 삭제 연산을 한 후에 변경할 내용을 쓸 수 있다. 게다가, 한 블록에 대한 삭제 연산의 횟수는 제한되어있다. 이러한 문제들을 해결하기 위해서, 운영체제는 FTL(Flash Translation Layer)이라는 디바이스 드라이버 소프트웨어를 사용한다. FTL 은 파일 시스템으로부터 읽기,쓰기 요청을 받고 논리 주소를 물리 주소로 변환한다.

본 논문에서는 DRAM/MRAM 하이브리드 메인 메모리와 플래시메모리 저장 장치를 지원하는 미래의 모바일 컴퓨터를 위한 버퍼 캐시 기법을 연구했다. (그림 1)은 본 논문의 시스템 구조를 보이고 있다. 제안한 버퍼 캐시 기법은 MRAM 의 쓰기 연산 성능과 플래시 메모리의 삭제 연산 횟수를 최소화하고, 기존의 버퍼 캐시 기법의 성능을 능가한다.

본 논문의 구성은 다음과 같다. 2 장에서는 플래시 메모리와 MRAM 의 특징을 설명하고 FTL 과 기존의 버퍼 캐시 기법을 소개한다. 3 장에서는 본 논문에서 제안하는 HAC 기법을 설명하고, 4 장에서는 제안한 버퍼 캐시 기법의 성능을 평가하고 실험 결과를 기술한다.

* 이 논문은 2010 년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(2010-0021897)



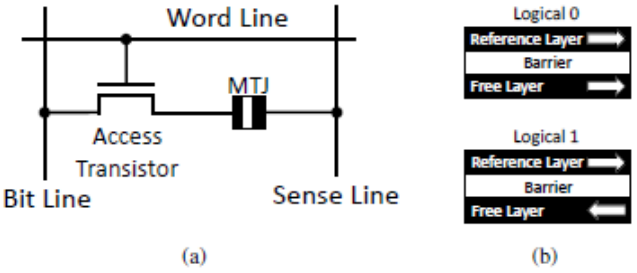
(그림 1) 시스템 구조

5 장에서는 본 논문의 결론을 맺는다.

2. 관련연구

2.1 MRAM

MRAM 의 셀은 이진 데이터(binary data)를 저장하는 자기터널접합(MTJ,Magnetic Tunnel Junction)을 사용한다. MTJ 는 참조층(reference layer)과 자유층(free layer)으로 이루어진 두 개의 강자성층(ferromagnetic)과 한 개의 터널 배리어층(tunnel barrier layer)으로 구성되어있다. 참조층의 자기 방향은 고정되어있는 반면, 자유층의 자기 방향은 셀에 저장된 이진 데이터에 따라 평행과 역 평행으로 나누어진다.



(그림 2) (a)MRAM 셀, (b)MRAM 의 병렬 MTJ

(그림 2)의 (a)는 MRAM 셀의 구조를 보여준다. DRAM 의 셀과 마찬가지로, MRAM 의 셀에는 저장 장치와 비트라인(bitline)을 연결하는 액세스 트랜지스터(access transistor)가 있다. 그러나, DRAM 과는 다르게 저장 장치의 다른 쪽 끝은 접지에 연결 되어있지 않고 센스선(sense line)에 연결되어있다.

MTJ 에서 데이터는 자유층의 자기 방향으로 저장된다. 이 방향은 셀에 저장된 데이터를 읽기 위해서 사용되는 장치의 전기 저항을 결정한다. (그림 2)의 (b)에서처럼, 자유층의 자기장과 참조층이 평행하면(즉, 같은 방향으로 정렬) MTJ 의 저항은 낮고, 논리는 0 이다. 반면, 자유층의 자기장과 참조층이 역 평행하면(즉, 반대 방향으로 정렬) MTJ 의 저항은 높고 논리는 1 이다. 셀에 저장된 데이터를 읽기 위해서, 센스선과 비트선 사이에서 매우 작은 전압이 소모되고, 전류 흐름의 양이 감지된다. 즉, MRAM 셀에 데이터를 쓰는 것은 전압 방식(voltage-mode)으로 작동하는 것이 아니라 전류 방식(current-mode)으로 작동한다. 특히, MTJ 에 데이터를 쓰기 위해서는 자유층의 자기 방향을 바꾸기 위해 많은 양의 전류가 MTJ 를 통해 들어가야만 한다. 전류의 방향에 따라, 자유층은 평행하거나 역 평행되어 고정층(fixed layer)이 된다. MTJ 에 데이터를 쓰기 위해서 요구되는 전류의 양은 읽기 위해서 요구되는 양보다 훨씬 더 많다.

2.2 NAND 플래시 메모리

NAND 플래시 메모리는 블록(block)들로 구성되며, 각 블록은 고정된 개수의 페이지(page)로 구성된다. NAND 플래시 메모리에 대한 기본 명령어는 읽기, 쓰기, 삭제이다. 읽기와 쓰기의 기본 단위는 페이지이며, 삭제는 블록 단위로 수행된다. 플래시 메모리는 데이터가 저장되어있는 블록을 사전에 삭제하지 않으면 변경할 수 없다. 한 블록에 대한 삭제 연산의 횟수는 제한되어있다. 삭제 연산은 오직 전체 블록에서만 수행된다. 운영체제는 쓰기 전 삭제(erase-before-write)문제를 해결하기 위해, FTL(Flash Translation Layer)이라는 디바이스 드라이버를 사용한다. FTL 은 삭제 연산 수를 줄이기 위해 논리 주소를 물리 주소로 변환한다. 대부분의 주소 변환 기법은 로그 블록 메커니즘(log block mechanism) 을 사용한다.

또한, 플래시 메모리를 고려한 버퍼 캐시 기법에 대한 연구가 있다. CFLRU(clean first least recently used)로 불리는 페이지 수준 기법이 제안되었다. CFLRU 는 LRU 순서에 의해 페이지 리스트를 유지한다. CFLRU 의 리스트는 LRU 리스트에서 상대적으로 최근에 참조된 페이지들을 가지고 있는 작업 영역(working region)과 상대적으로 오래 전에 참조된 페이지들을 가지고 있는 클린-우선 영역(clean-first region)으로 나누어진다. 클린(clean) 페이지란 내용이 변경되지 않은 페이지를, 더티(dirty) 페이지는 내용이 변경된 페이지를 뜻한다. 쓰기 연산의 비용을 줄이기 위해서, CFLRU 는 버퍼 교체 시에 LRU 리스트의 끝에서부터 클린-우선 영역의 클린 페이지를 찾아 교체한다. 만약 클린-우선 영역에 클린 페이지가 없다면 LRU 리스트의 맨 마지막 더티 페이지를 교체한다. CFLRU 는 페이지 캐시에 더티 페이지의 flush(교체)를 지연시켜 쓰기와 삭제 연산의 수를 줄일 수 있다.

블록 수준 기법은 버퍼를 플래시 메모리의 삭제 블록 단위로 관리하며, 버퍼 교체 시에 블록을 선택하여 소속된 페이지를 모두 교체시킨다. BPLRU(Block Padding LRU)는 플래시 메모리 블록을 기반으로 LRU 리스트를 관리한다. 버퍼 캐시에 페이지가 참조될 때마다, 같은 블록의 모든 페이지는 MRU(Most Recently Used) 위치로 이동된다. 버퍼 캐시가 가득 차면, BPLRU 는 희생 블록의 모든 페이지를 교체한다. 게다가, 저장되는 페이지는 FTL 이 로그 블록에 저장하므로 BPLRU 는 스위치 병합을 하도록 페이지 패딩(page

padding) 기법을 제안했다. 페이지 패딩 기법은 희생 블록의 페이지 중에서 버퍼에 없는 페이지를 플래시 메모리에서 버퍼로 읽고, 블록의 모든 페이지를 한번에 플래시 메모리에 저장시키는 기법이다. BPLRU 에서 모든 로그 블록은 스위치 병합을 하기 때문에 삭제 연산의 수를 줄일 수 있다.

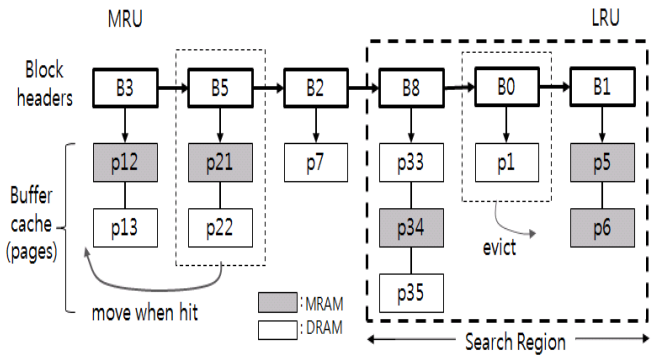
3. HYBRID MEMORY AWARE CACHING

본 논문에서 새로운 버퍼 캐시 기법 HAC(Hybrid memory Aware Caching)를 제안한다. HAC 는 (그림 3)과 같이 플래시 메모리의 블록을 기반으로 LRU 리스트를 관리한다. LRU 리스트는 플래시 메모리에서 로드 된 자신의 페이지를 관리하는 각 블록의 헤더로 구성되어있다.

플래시 메모리의 블록 B 의 페이지 p 가 처음 참조될 때, HAC 는 새로운 버퍼를 할당하고 할당된 버퍼에 페이지 p 를 저장한다. 만약 블록 B 의 블록 헤더가 존재하지 않으면, HAC 는 새로운 블록 헤더를 할당하고 LRU 리스트의 MRU 위치에 놓는다. 그리고는, HAC 는 블록 B 의 헤더에 페이지 p 의 버퍼를 연결한다. 버퍼 캐시에 페이지가 참조될 때마다, 같은 블록에 모든 페이지는 MRU 위치로 옮겨진다.

우리는 메인 메모리가 메모리 주소에 의해 DRAM 과 MRAM 으로 나누어져 있다고 가정한다(Seok, 2012). 낮은 메모리 주소는 DRAM 으로 할당되고, 높은 메모리 주소는 MRAM 으로 할당된다. HAC 가 새로운 버퍼를 할당할 때, 낮은 섹션에 우선적으로 할당한다. 또한, HAC 는 각 블록의 메모리 유형(DRAM 또는 MRAM)을 유지하고 블록에 동일한 메모리 유형의 버퍼를 할당하려고 시도한다. DRAM 버퍼를 할당하려고 할 때 free DRAM 버퍼가 없다면, HAC 는 LRU 리스트의 검색 영역(search region)에서 클린 DRAM 버퍼를 찾고 이를 free 시킨다.

HAC 는 시스템 내의 프리 버퍼가 여전히 이용 가능함에도 불구하고, 프리 클린 DRAM 버퍼를 우선적으로 반납하는 기법을 제안한다. 대규모 메인 메모리에서 머지않아 액세스 되지 않는 사용 버퍼가 많이 있기 때문에, 캐시 성능에 적은 영향으로 버퍼를 프리 시킬 수 있다.



(그림 3) HAC 의 LRU 리스트

HAC 는 주기적으로 검색 영역에서 오직 DRAM 버퍼만 사용하는 클린 블록을 찾거나, 프리 DRAM 버퍼의 수가 임계치 보다 적을 때 클린 블록을 찾는다. 그리고는, 블록을 free 시킨다. 이 기법은 HAC 가 새로운 할당을 위한 프리 DRAM 버퍼를 확보할 수 있기 때문에, MRAM 의 쓰기 연산의 수를 줄일 수 있다.

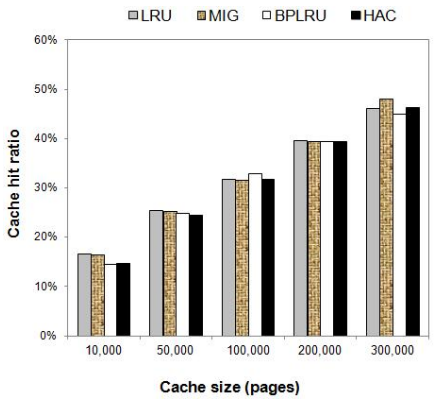
MRAM 의 쓰기 연산의 수를 더 줄이기 위해서, STT-RAM 의 클린 페이지가 쓰기연산에 의해 참조 되었을 때, HAC 는 DRAM 버퍼를 할당하고 DRAM 버퍼에 요청된 데이터를 쓴다.그리고는, MRAM 버퍼를 반납한다. 만약 프리 DRAM 버퍼가 없을 때, HAC 는 검색 영역의 클린 DRAM 버퍼를 프리로 만들고 이 버퍼를 요청된 쓰기 데이터를 저장하기 위해 사용한다.

만약 모든 버퍼가 사용 중이면, HAC 는 검색 영역에서 희생 블록을 선택한다. 플래시 메모리의 삭제 연산의 수를 줄이기 위해서, HAC 는 클린 블록을 찾고 그 블록의 모든 페이지를 프리로 만든다. 만약 검색 영역에 클린 블록이 없다면, HAC 는 페이지 패딩 기법을 수행해서 블록 리스트의 LRU 위치에 있는 희생 블록을 선택하고, 희생 블록의 모든 페이지를 플러시한다.

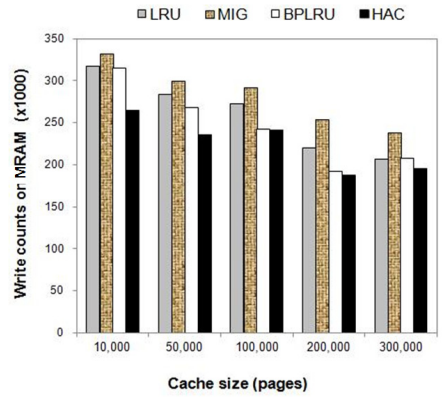
4. 실험 결과

본 논문에서 제안한 HAC 의 성능을 검증하기 위해 트레이스 기반의 시뮬레이터를 구현했다. 모바일 컴퓨터의 워크로드를 수집하기 위해서, 일주일 동안 노트북 PC 에 여러 응용 프로그램을 실행시켜 디스크 I/O 를 수집했다. 총 I/O 횟수는 706,833 이고 읽기/쓰기 비율은 55:45 이다.

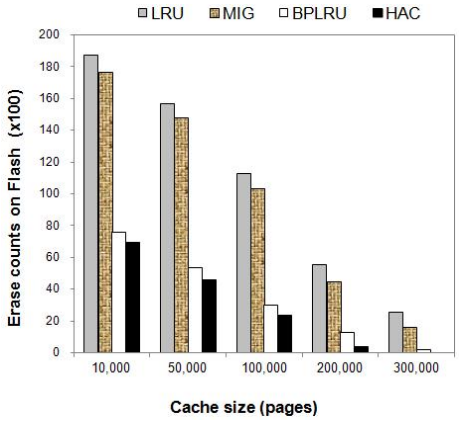
버퍼 캐시 크기를 10,000 개에서 300,000 개로 변화시키면서 버퍼 캐시 적중률, MRAM 의 쓰기 연산 횟수, 플래시의 삭제 연산 횟수를 측정했다. 그림 1 은 각 버퍼 교체 기법에 대해 버퍼 캐시의 크기를 변화시켰을 때 캐시 적중률을 보여준다. 그림 2 는 각 버퍼 교체 기법에 대해 MRAM 의 쓰기 연산 횟수를 보여준다. HAC 는 쓰기 연산의 횟수를 평균적으로 약 13%, 최대 26%까지 줄인다. (그림 4)에서처럼 HAC 는 BPLRU 만큼이나 플래시메모리의 삭제 연산 횟수를 줄일 수 있다. 게다가 HAC 는 교체 절차 동안 삭제 연산을 피하기 위해 클린 블록을 고려하기 때문에 BPLRU 의 성능을 능가한다.



(a) 버퍼 캐시 적중률



(b) MRAM의 쓰기 연산 횟수



(c) 플래시메모리의 삭제 연산 횟수

(그림 4) 성능 평가 결과

5. 결론

MRAM 이나 플래시메모리 같은 저 전력 비 휘발성 메모리는 모바일 컴퓨터에서 사용될 차세대 메모리로 각광 받고 있다. 제안된 버퍼 캐시 기법은 DRAM/MRAM 하이브리드 메인 메모리와 플래시 메모리 저장장치를 지원한다. 트레이스 기반 시뮬레이션을 통해 HAC 가 기존의 버퍼 캐시 기법의 성능을 능가하는 것을 증명했다.

참고문헌

[1] H. Jang, B. An, N. Kulkarni, K. Yum, E. Kim, "A Hybrid Buffer Design with STT-MRAM for On-Chip Interconnects," in Proc. of ACM/IEEE International Symposium on Networks-on-Chip (NOCS), 2012.

[2] S. Park, S. Gupta, N. Mojumder, A. Raghunathan, and K. Roy, "Future Cache Design using STT MRAMs for Improved Energy Efficiency: Devices, Circuits and Architecture," in Proc. of Design Automation Conference (DAC), pp. 492-497, 2012.

[3] C. Smullen, V. Mohan, A. Nigam, S. Gurumurthi, and M. Stan, "Relaxing non-volatility for fast and energy-efficient STT-RAM caches," in Proc. of High Performance Computer Architecture (HPCA), pp. 50-61, 2011.

[4] E. Kultursay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu, "Evaluating STT-RAM as an Energy-Efficient Main Memory Alternative," in Proc. of IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2013.

[5] A. Charles, N. Mojumder, X. Fong, S.Choday, S. Park, and K. Roy, "Spin-Transfer Torque MRAMs for Low Power Memories: Perspective and Prospective," IEEE Sensors, Vol. 12, No. 4, pp. 756-766, 2012.

[6] Barroso, L., Holzle, U., 2007. The Case for Energy-proportional Computing. *Computer*, Vol.40, No.12.

[7] Qureshi, M., Srinivasan, V., Rivers, J., 2009. Scalable High Performance Main Memory System Using Phase-Change Memory Technology. In *Proceedings of International Symposium on Computer Architecture*.

[8] Park, H., Yoo, S., Lee, S., 2011. Power Management of Hybrid DRAM/PRAM-based Main Memory. In *Proceedings of Design Automation Conference*.

[9] Kim, J., Noh, S., Min, S., Cho, Y., 2002. A Space-Efficient Flash Translation Layer for Compactflash Systems. *IEEE Transactions on Consumer Electronics*, Vol. 48, No. 2, pp. 366-375.

[10] Ryu, Y., 2010. SAT: Switchable Address Translation for Flash Memory Storages. In *Proceedings of IEEE Computer Software and Applications Conference.*

[11] Park, S., Jung, D., Kang, J., Kim, J., Lee. J., 2006. CFLRU: A Replacement Algorithm for Flash Memory. In *Proceedings of International Conference on Compilers, Architecture and Synthesis for Embedded Systems*.

[12] Kim, H., Ahn, S., 2008. BPLRU: A Buffer Management Scheme for Improving Random Writes in Flash Storage. In *Proceedings of the 6th USENIX Conference on File and Storage Technologies*.

[13] Seok, H., Park, Y., Park, K., Park, K., 2012. Efficient Page Caching Algorithm with Prediction and Migration for a Hybrid Main Memory. *Applied Computing Review*, Vol. 11, No. 4.