IPSiNS: 낸드 플래시 메모리 기반 저장 장치를 위한 입출력 성능 시뮬레이션 도구

(IPSiNS: I/O Performance Simulation Tool for NAND Flash Memory - based Storage System)

윤 경 훈 † 정 호 영 †
(Kyeong-hoon Yoon) (Ho-young Jung)
박 성 민 † 심 효 기 †
(Sung-min Park) (Hyo-gi Sim)
차 재 혁 †† 강 수 용 †††
(Jae-hyuk Cha) (Sooyong Kang)

요 약 낸드 플래시 메모리 기반 저장장치를 블록 디바이스로 가상화하는데 사용되는 플래시 변환 계층(FTL)는 플래시 메모리의 특성만을 고려하여 설계되었다. 그러나 FTL에서는 운영체제의 버퍼교체정책을 거쳐 발생되는 입출력 요청을 처리하기 때문에, 버퍼교체정책과 FTL 알고리즘은 큰 연관성을 가지게 된다. 따라서, 버퍼교체정책과 FTL 알고리즘을 동시에 고려하지 않고서는 플래시메모리 기반 저장장치를 사용하는 시스템의 전체적인 입출력 성능을 평가할 수 없으므로, 그 두 요소를 동시에 고려한 버퍼교체정책 또는 FTL 알고리즘에 대한 연구가 필요하다. 본 연구에서는 그러한 통합연구에 사용될 수 있는 입출력 성능 평가 도구인 IPSiNS를 개발하였다.

· 본 연구는 한국과학재단 특정기초연구(R01-2006-000-10630-0)지원으로 수행되었음
· 이 논문은 2007 한국컴퓨터종합학술대회에서 'IPSiNS: 낸드 플래시 메모리 기반 저장 장치를 위한 입출력 성능 시뮬레이션 도구'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 한양대학교 전자컴퓨터통신공학과
rainynova@gmail.com
horong@hanyang.ac.kr
syrilo@hanyang.ac.kr
dahlia@hanyang.ac.kr
†† 종신회원 : 한양대학교 정보통신학부 교수
chajh@hanyang.ac.kr
††† 종신회원 : 한양대학교 컴퓨터교육과 교수
sykang@hanyang.ac.kr
(Corresponding author임)

논문접수 : 2007년 10월 5일
심사완료 : 2007년 10월 23일

키워드 : 플래시 메모리, 입출력 성능, 시뮬레이션 도구

Abstract Flash Translation Layer(FTL) which enables NAND Flash memory-based storage system to be used as a block device is designed considering only characteristics of NAND Flash memory. However, since FTL processes I/O requests which survived against buffer replacement algorithm, FTL algorithm has tight relationship with buffer replacement algorithm. Therefore, if we do not consider both FTL and buffer replacement algorithms, it is difficult to predict the actual I/O performance of the computer systems that have Flash memory-based storage system. The necessity of FTL and buffer replacement algorithm co-design arises here. In this work, we implemented I/O performance evaluation tool, IPSiNS, which simulates both the buffer replacement and FTL algorithms, simultaneously.

Key words : Flash memory, I/O Performance, Simulation tool

1. 서 론

현재 낸드[1] 플래시 메모리 기반 저장 장치를 파일시스템에서 접근할 수 있는 블록 장치로 사용하기 위하여 플래시 변환 계층(이하 FTL)가 널리 사용된다.

FTL의 기능 중 맵핑 알고리즘은 입출력 요청의 패턴에 따라 서로 다른 수의 지우기/쓰기 연산을 발생시키게 된다. 따라서, FTL에 보내지는 입출력 요청의 패턴이 FTL 맵핑 알고리즘의 특성에 맞게 조절될 수 있다면, 시스템의 전체적인 입출력 성능을 향상시킬 수 있게 된다. 바로 이 점에서, FTL 에 가해지는 입출력 요청 패턴을 최종적으로 결정하는 버퍼교체정책과 FTL 알고리즘의 연관성이 발생하게 된다. 즉, 어떤 버퍼 교체 정책을 사용하는지에 따라 낸드 플래시 메모리 기반 저장 장치의 전체적인 성능이 달라질 수 있게 된다.

그러므로, 플래시 메모리 기반 저장장치의 전체적인 성능을 향상시키기 위해서는 FTL의 맵핑 알고리즘과 메모리 관리계층의 버퍼교체 알고리즘이 상호 보완적으로 동작하도록 두 알고리즘을 연계하여 설계하여야 한다.

본 논문에서는, 플래시 메모리 기반 저장 장치를 위한 버퍼 교체 정책과 FTL 맵핑 알고리즘의 통합 연구를 지원하기 위해 개발된 입출력 시뮬레이션 및 성능 분석 도구 IPSiNS(I/O Performance Simulation Tool for NAND Flash Memory - based Storage System)를 소개한다. IPSiNS는 응용프로그램이 발생시키는 입출력

1) 이후 본 논문에서 표현되는 "플래시 메모리"는 모두 "낸드 플래시 메모리"를 말한다.

요청 트레이스를 입력 받아 각각의 요청이 버퍼 교체 정책과 FTL의 맵핑 알고리즘을 거쳐서 최종적으로 플래시 메모리에 전달되는 과정을 시뮬레이션 하여 결과적으로 플래시 메모리에서 수행되는 읽기/쓰기/지우기 연산의 횟수를 산출하고, 그 모든 연산들이 수행되는데 걸리는 시간은 물론, 버퍼에서의 읽기/쓰기 히트 관련 정보, FTL 맵핑 알고리즘이 유발시키는 읽기/쓰기/지우기 연산의 발생과 관련된 정보들을 생산한다. 이러한 정보들은 플래시 메모리 기반 저장 장치의 성능에 영향을 미치는 각 요소들에 대한 면밀한 분석에 필수적이며, 그러한 분석을 통해 더 나은 버퍼 교체 정책과 FTL 맵핑 알고리즘을 개발할 수 있다. 또한, 본 논문에서는 IPSiNS를 이용하여 기존의 버퍼 교체 알고리즘과 FTL 맵핑 알고리즘이 플래시 메모리 기반 저장 장치의 성능 측면에서 어떠한 연관 관계를 가지는지 분석하였다.

2. 연구의 필요성

운영체제의 버퍼 교체 정책으로는 FAB, LIRS, ARC 등이 있으며[2-4], FTL 알고리즘으로는 BAST, FAST, SSR, Mitsubishi, FMAX, LEXAR 등이 있다[5-9].

플래시 메모리가 많이 사용됨에 따라 입출력 성능을 향상을 위해 플래시 메모리 특성을 고려한 많은 연구들이 진행되고 있다. FAB[2]는 플래시 메모리의 지우기 단위가 블록 단위인 것을 고려하여 지우기를 줄이고자 버퍼 캐쉬에서 블록 단위로 페이지를 묶어서 요청하는 버퍼 교체 정책 알고리즘이다. BAST[5]와 FAST[6]는 다수의 데이타 블록과 소수개의 로그 블록으로 구성되는 로그 블록 기법(Log Block Scheme)을 사용한다.

입출력 관련 알고리즘의 성능을 평가하기 위해서는 입출력 트레이스가 필요하다. 입출력 트레이스란 물리적인 저장장치에 요청되는 LBA(Logical Block Address)와 요청의 종류(읽기/쓰기) 쌍의 집합으로 볼 수 있으며, 그것이 추출되는 계층에 따라서 버퍼 입출력 트레이스와 디스크 입출력 트레이스로 구분된다. 버퍼 입출력 트레이스는 파일시스템에 요청되는 입출력의 집합으로서, 버퍼 알고리즘의 성능을 실험 하기 위해 사용될 수 있고, 디스크 입출력 트레이스는 파일시스템이 물리적인 저장장치(하드디스크, 플래시 메모리 등)에 요청되는 입출력의 집합으로서, FTL의 맵핑 알고리즘의 성능을 평가하기 위해 사용될 수 있다.

메모리 관리계층의 버퍼교체 알고리즘에 대한 성능 측도로는 히트 율이 사용되며, FTL 맵핑 알고리즘에 대한 성능 측도로는 쓰기/지우기 횟수가 사용된다. 서로 다른 성능 측도에 맞게 설계된 정책들이 한 시스템 내부에서 상호 연관성을 가지는 상황에서 사용될 경우, 시스템의 전체적인 성능은 각 정책의 성능에 반드시 의존하지 않을 수 있다. 예를 들어, 버퍼교체 알고리즘의 히트 율이 높으면, 플래시 메모리에 대한 입출력 요청이 줄어들기 때문에, 플래시메모리에서의 쓰기/지우기 횟수가 줄어들 것이라 예상할 수 있지만, 실제로는 정 반대의 현상이 생길 수도 있다. 본 연구에서 개발된 IPSiNS를 이용한 실험 결과(표 1)는 그러한 현상을 확인시켜 준다. (실험과 관련된 자세한 내용은 4장에 기술되어 있다.)

표 1 BAST를 사용한 LRU와 FAB의 성능 비교

	운영체제 계층	낸드 플래시 메모리 계층	
	버퍼 캐시 히트율	쓰기 횟수	지우기 횟수
LRU	21.44 %	5313	188
FAB	20.14 %	4407	145

표에서 볼 수 있듯이, 버퍼에서의 히트 율은 LRU가 FAB보다 1.3% 더 높지만, 플래시메모리에서 발생하는 쓰기/지우기 횟수는 LRU가 FAB보다 각각 906번, 43번이 더 많다는 것을 알 수 있다. 앞서 기술했듯이 버퍼교체정책의 성능 측도로 볼 때는 LRU가 FAB에 비해 더 좋은 성능을 보이지만, 플래시 메모리에서의 성능 측도인 쓰기와 지우기 횟수를 기준으로 볼 때는 FAB가 LRU 보다 더 좋은 성능을 보이고 있다. 따라서, 다양한 알고리즘의 입출력 성능을 비교 평가하기 위해서는 통합된 성능 측도가 필요하고, 그 측도를 기준으로 한 높은 성능의 알고리즘이 개발되어야 한다. 본 연구에서는 그러한 통합된 성능 측도로 "수행 시간(Execution Time)"을 사용했다. 수행시간은 입출력 요청을 발생시키는 프로그램이 각 입출력 처리가 끝날 때까지 대기하는 시간의 합을 말한다. 수행시간은 삼성 플래시 메모리 데이타시트[1]를 근거로 플래시 메모리의 읽기 수행 시간을 1로 봤을 때, 상대적으로 쓰기는 10배, 지우기는 80배로 계산 하였다. 수행 시간을 기준으로 두 알고리즘의 성능을 측정한 결과, FAB가 LRU 보다 11.2% 더 좋은 성능을 보이는 것으로 나타났다.

이 실험 결과를 통해 알 수 있듯이, 플래시메모리 기반 저장장치를 사용하는 시스템에서의 버퍼교체정책은 FTL 알고리즘을 고려하여야 하며, 반대로 FTL 알고리즘을 설계하기 위해서는 버퍼교체정책을 고려하여야 한다. 궁극적으로는 버퍼교체정책과 FTL 알고리즘의 통합 설계(co-design)가 필요하게 된다. 따라서, 그러한 통합 설계를 위해서는 한 부분의 동작이 다른 부분에 어떤 영향을 미치는지를 파악할 수 있어야 하고, 통합 설계를 통해 개발된 기법이 전체적으로 어떤 성능을 보이는지 평가할 수 있어야 한다. 이를 위해서 버퍼교체정책과 FTL의 맵핑 알고리즘을 동시에 고려한 성능평가 도구의 개발이 필요하게 된다.

3. IPSiNS의 설계와 구현

그림 1에서는 입출력 트레이스 파일을 지정하고, 버퍼 교체알고리즘과 FTL 알고리즘을 선택하며, 각 알고리즘의 파라메터를 설정한다.

버퍼 시뮬레이션 계층은 트레이스를 입력 받아 선택된 버퍼 교체 알고리즘에 따라 Hit/Miss를 결정하고, 필요할 경우 페이지 교체를 함으로써 버퍼의 상태를 변화시킨다.

플래시 메모리 맵핑 알고리즘 계층에서는 버퍼 교체 정책 계층에서 미스가 발생됨에 따라 요청되는 입출력의 트레이스를 받아서, 사용자 인터페이스 계층에서 선택된 FTL 알고리즘의 동작 방식에 따라 발생되는 플래시메모리에서의 읽기/쓰기/지우기 연산의 순서를 생성한다. 여기서 생성된 연산 순서는 시뮬레이션 결과 표현계층에서 그래프와 다양한 통계 수치(버퍼 캐시 히트 율, 플래시 메모리 읽기/쓰기/지우기 횟수 등)로 표현된다.

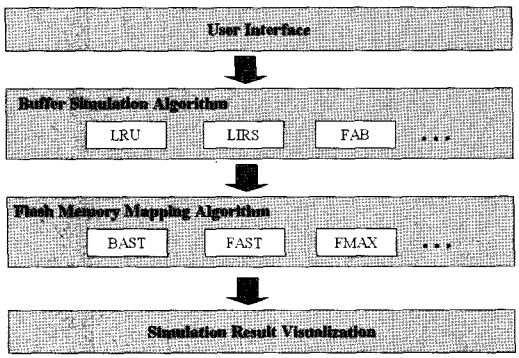

그림 1 IPSiNS의 소프트웨어 구조

IPSiNS는 MS 윈도우즈 운영체제에서 동작되며, C/C++/MFC 언어를 사용하여 비주얼 스튜디오 닷넷 2003으로 개발하였다. 그림 2는 IPSiNS의 전체적인 UI를 보여주고 있다. 그림에서 A는 입출력 트레이스에 대한 메타 정보를 표시하는 영역이고, B는 영역 E에 나타나는 그래프의 표현 형식을 설정하는 부분이다. C는 버퍼 교체 알고리즘과 파라메터를 설정하고, 시뮬레이션이 끝난 후 버퍼교체 알고리즘 만의 성능(히트 율)을 보여주는 영역이다. D는 FTL 알고리즘과 파라메터를 설정하고, 시뮬레이션이 끝난 후 플래시메모리에서 행해진 각종 연산의 수와 통합측도인 수행시간을 표시하는 부분이다. 마지막으로 E는 각종 그래프가 그려지는 영역이다. 영역 E에 그려지는 그래프는 버퍼크기에 따른 히트 율 그래프와, 버퍼에서의 페이지 접근 패턴을 그린 Hit/Miss 그래프가 있다. 히트 율 그래프는 영역 C에서 복수개의 알고리즘이 선택되었을 때 선택된 모든 알고리즘의 히트 율을 동시에 보여줌으로써 알고리즘간 상호 비교가 가능하게 한다. 예를 들어, 그림 2의 영역 E에 그려진 그래프는 FTL 알고리즘으로 BAST를 사용했을 때, LRU, FAB, LIRS 버퍼교체 알고리즘의 히트 율을 버퍼 크기에 따라 그린 것이다.

4. 실 험

이 장에서는 IPSiNS를 사용하여 버퍼교체 알고리즘이 플래시메모리 기반 저장장치를 사용하는 시스템의

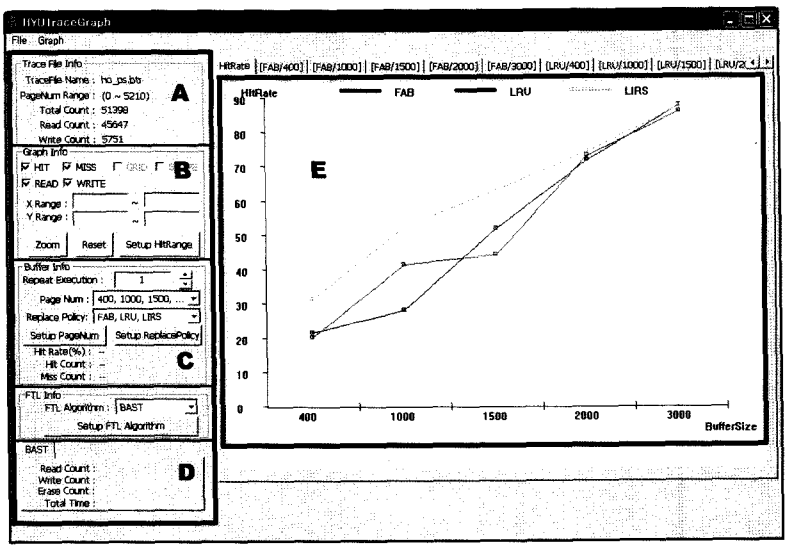

그림 2 IPSiNS의 그래픽 사용자 인터페이스

표 2 시뮬레이션 파라메터

버퍼 트레이스 파일	위스콘신 벤치마크 의 PostgreSQL (읽기 56475, 쓰기 5751)
버퍼 페이지 개수	500, 1000, and 1500
버퍼 교체 정책	FAB, LRU, and LIRS
FTL 맵핑 알고리즘	BAST (로그 블럭 수 : 64)

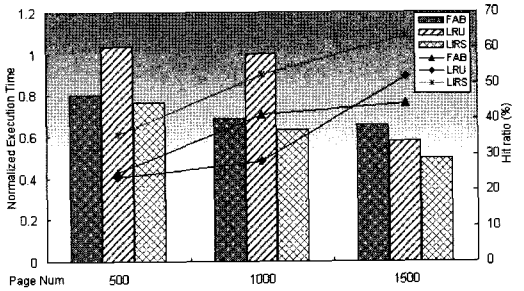

그림 3 히트율과 실행시간과의 관계

입출력 성능에 미치는 영향에 대해 기술한다. 이를 위해 먼저 표 2에 기술된 바와 같이 IPSiNS를 설정하고 시뮬레이션을 수행하였다.

그림 3은 버퍼 히트 율과 실행시간과의 관계를 나타낸 것이다. 막대 그래프는 페이지 수가 1,000일 때 LRU의 실행시간을 1로 두고, 각 버퍼교체 알고리즘의 상대적인 실행시간을 그린 것이고, 꺾은 선 그래프는 각 알고리즘의 히트 율을 나타낸다. 페이지 개수가 500개일 때, FAB와 LRU는 비슷한 히트 율을 보이지만, 실행시간의 차이는 크게 나타나고 있다. 실제로 FAB의 실행시간은 히트 율이 10%가량 더 높은 LIRS와 비슷하게 나타난다. 히트 율과 실행시간의 이러한 관계는 페이지 개수가 1,000일 때도 나타난다. 따라서, 이 실험을 통해 버퍼 히트 율과 입출력 실행시간은 비례관계가 아니며, 히트 율을 최대화하는 것이 시스템의 입출력 성능을 최대화하지 않을 수 있다는 사실을 확인할 수 있다.

그림 4는 플래시메모리에서 발생하는 쓰기 연산의 수와 실행시간과의 관계를 나타낸 것이다. 그림에서 볼 수 있듯이, 쓰기 연산의 횟수가 적어지면 실행시간 역시 그에 비례하여 줄어든다는 사실을 알 수 있다. 즉, 플래시메모리 기반 저장장치를 사용하는 시스템의 입출력 성능은 쓰기 연산의 횟수에 크게 의존한다고 할 수 있다.

FTL 맵핑 방식에 따른 지우기 연산의 횟수 변화를 알기 위하여 표 3과 같이 IPSiNS를 설정하고 시뮬레이션을 수행하였다.

그림 5는 각 버퍼 교체 정책에 대해 서로 다른 FTL 맵핑 방식을 사용했을 때 지우기 연산의 횟수를 그린 그래프이다. 같은 블록에 속한 페이지들을 순차적으로

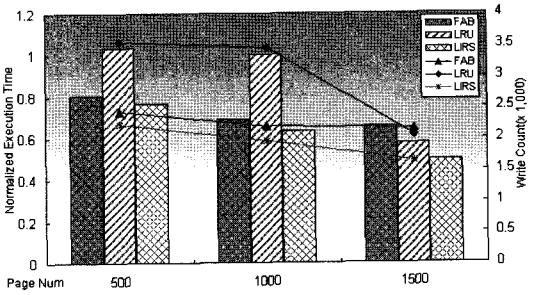

그림 4 쓰기 요청 횟수와 실행시간과의 관계

표 3 IPSiNS 실험 환경(GCC)

버퍼 트레이스 파일	GCC (읽기 139579, 쓰기 19088)
버퍼 페이지 개수	2000
버퍼 교체 정책	FAB and LRS
FTL 맵핑 알고리즘	BAST and FAST (로그 블럭 수 : 64)

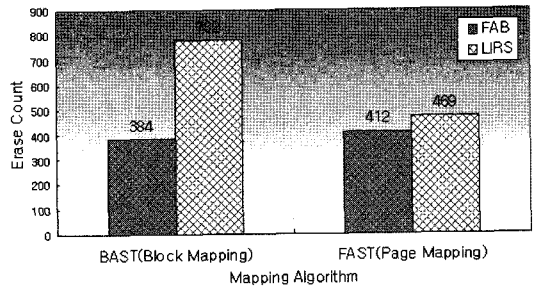

그림 5 맵핑 방식과 Erase Count와의 관계

플래시메모리에 쓰는 경향을 보이는 FAB의 경우, 1:1 블록 맵핑 방식인 BAST를 사용할 때가 M:N블록 맵핑 방식인 FAST를 사용할 때 보다 적은 수의 지우기 연산을 발생시켰으며, 연속적이지 않은 임의의 페이지 단위 쓰기를 발생시키는 경향이 있는 LIRS의 경우, FAST를 사용할 때가 BAST를 사용할 때 보다 적은 수의 Erase 연산을 발생시키는 것을 알 수 있다. 이 결과를 볼 때, 버퍼 교체 정책과 FTL 맵핑 알고리즘을 설계할 때는 버퍼교체 정책이 발생시키는 쓰기 연산의 패턴과 FTL 맵핑 알고리즘의 성격을 모두 고려해야 할 것이다.

5. 결 론

본 연구에서 개발한 IPSiNS는 플래시메모리 기반 저장장치를 사용하는 시스템을 위한 버퍼교체 정책과 FTL 알고리즘을 통합 설계하는데 유용하게 활용될 수 있다. IPSiNS는 운영체제의 버퍼교체 정책과 FTL 맵핑 알고리즘을 연계하여 시뮬레이션 함으로써, 플래시 메모리를

저장장치로 사용하는 컴퓨터 시스템의 입출력 성능을 보다 정확하게 예측할 수 있으며, 그에 따라 시스템에 적합한 버퍼교체 알고리즘과 FTL 맵핑 알고리즘의 조합을 찾을 수 있다.

참 고 문 헌

[1] K9K8G08U1M(1G x 8 Bit NAND Flash Memory) Data Sheet, http://www.samsung.com/products/semiconductor/NANDFlash/SLC_LargeBlock/8Gbit/K9K8G08U1M/K9K8G08U1M.htm, Samsung Electronics Co. Ltd.

[2] Heeseung Jo, Jeong-Uk Kang, Seon-Yeong Park, Jin-Soo Kim, and Joonwon Lee, FAB: Flash-Aware Buffer Management Policy for Portable Media Players, IEEE 2006.

[3] S. Jiang and X. Zhang, "LIRS: An Efficient Low Inter-reference Recency Set Replacement Policy to Improve Buffer Cache Performance, In Proceeding of 2002 ACM SIGMETRICS," June 2002, pp. 31-42.

[4] Nimrod Megiddo, and Dharmendra Modha, "ARC: A Self-Tuning, Low Overhead Replacement Cache," Proc. 2nd USENIX Conference on File and Storage Technologies (FAST 03), 2003.

[5] Jesung Kim, Jong Min Kim, Sam H. Noh, Sang Lyul Min and Yookun Cho, A SPACE-EFFICIENT FLASH TRANSLATION LAYER FOR COMPACTFLASH SYSTEMS, IEEE Transactions on Consumer Electronics, Vol. 48, No. 2, MAY 2002.

[6] Sang-Won Lee, Dong-Joo Park, Tae-Sun Chung, Dong-Ho Lee, Sangwon Park, Ha-Joo Song, "A Log Buffer based Flash Flash Translation Layer using Fullly Associative Sector Translation," ACM Transactions on Embedded Computing Systems.

[7] Bum Soo Kim And Gui Young Lee, "Method of Driving Remapping in Flash Memory and Flash Memory Architecture Suitable Threfore," United States Patent, No. 6,381,176, 2002.

[8] Takayuki Shinohara, "Flash Memory Card with Block Memory Address Arrangement," United States Patent, No. 5,905,993, 1999.

[9] Petro Estakhri and Berhanu Iman, "Moving Sequential Sectors within A Block of Information in A Flash Memory Mass Storage Architecture," United States Patent, No. 5,930,815, 1999.