Solid-State Disk의 빠른 구조 탐색 및 최적화를 위한 상위 수준 시뮬레이션 환경

*방관후, 정의영
연세대학교 전기전자공학과
e-mail : khbang@dtl.yonsei.ac.kr, eychung@yonsei.ac.kr

A High-level Simulation Environment for Architecture Exploration and Optimization of Solid-State Disk

*Kwanhu Bang, Eui-Young Chung
School of Electrical and Electronic Engineering
Yonsei University

Abstract

Recently, flash memory-based Solid-State Disk (SSD) is being spotlighted as a large-sized and high-performance storage device. In order to further improve the performance of SSD, it is required to explore the internal architecture of it as well as efficient algorithms for Flash Translation Layer (FTL). In this paper, we constructed a high-level SSD hardware platform using SystemC so that fast architecture exploration is possible. In addition, a variety of FTL algorithms can be evaluated quantitatively on the platform. Using our environment, both hardware and software part of SSD can be optimized fast.

I. 서론

다양한 전자기기들을 이용하여 실시간으로 대용량의 정보를 처리하고 있는 현대 사회에서는 이러한 정보의 안전하고 효과적인 저장과 빠른 접근에 대한 수요가 그 어느 때보다도 더 크다. 과거 수십 년간 이러한 정보 저장을 위해 널리 사용되었던 저장 장치인 하드 디스크 드라이브 (HDD)와 함께 최근에는 플래시 메모리 기반의 Solid-State Disk (SSD)가 등장하면서 대용량의 고성능 저장 장치의 개발이 가속화되고 있다.

SSD는 플래시 메모리와 같은 비휘발성의 반도체만으로 이루어져있어서 기존의 HDD와 같은 추가의 기계적인 구성 요소를 가질 필요가 없으며, 전기적 신호를 이용하여 빠른 접근이 가능하다. 또한 소비 전력이나 발열, 외부 충격 등에 있어서도 HDD보다 우월한 성능을 보여주고 있다. 따라서 최근에는 더욱 고성능의 SSD가 등장하고 있으며, 향후 HDD를 완전히 대체하는 새로운 저장 장치의 표준이 될 것이라 예측된다.

SSD는 대용량의 플래시 메모리 소자들을 단순하게 연결한 것이 아니라 이들 메모리를 효과적으로 접근하기 위해 SSD 제어기를 중심으로 설계된다. 또한 SSD 제어기 내부에서는 플래시 메모리의 고유 특성을 반영하여 Flash Translation Layer (FTL)라고 부르는 소프트웨어까지 탑재된다. 따라서 SSD 자체가 시스템 수준에서 볼 때 하나의 커다란 시스템이라 할 수 있다. SSD의 성능 향상을 위해서는 이러한 SSD 시스템의 내부 구조에 대한 탐색과 함께 소프트웨어와 하드웨어의 효율적인 분할 설계 및 각각의 최적화 등의 작업이 반드시 필요하다.

본 논문에서는 SSD의 구조 탐색을 빠르게 수행할 수 있는 하드웨어 플랫폼을 개발하고 이를 활용하여 다양한 SSD의 구동 알고리즘에 대한 정확한 성능 평

가를 할 수 있는 시뮬레이션 환경을 구축하였다. 빠른 구조 탐색을 위하여 하드웨어 플랫폼은 상위 수준 하드웨어 모델링 언어인 SystemC[5]를 이용하여 개발되었으며, FTL 소프트웨어는 주어진 하드웨어 제약 조건이 모두 반영되어 구동되기 때문에 알고리즘 자체에서는 고려하지 않고 있는 여러 변수들을 반영한 정확한 성능 평가 및 개선이 가능하다.

II. 본론

2.1 SSD의 성능 향상을 위한 관련 연구들

SSD는 내부에 FTL을 구동할 수 있는 마이크로컨트롤러와 이를 위한 메모리, 데이터를 저장하는 플래시 메모리 등 다양한 구성 요소들을 가지고 있다. 또한 이러한 구성 요소들이 동일한 구조로 구성되어 있는 경우에도 FTL의 알고리즘에 따라 성능의 차이를 보여준다. 따라서 효과적인 SSD의 성능 향상을 위한 여러 시도가 있었다.

SSD의 성능 향상을 위한 하드웨어 관점에서의 연구는 데이터 접근 및 저장의 병렬성을 극대화하기 위한 노력이 주를 이루었다. SSD 내부에서 사용되는 여러 개의 플래시 메모리들은 SSD 제어기에 복수의 채널을 사용하여 연결되며, 각 채널에는 여러 개의 플래시 메모리가 연결되어 동시에 여러 개의 플래시 메모리를 구동할 수 있다. 이러한 구조를 기반으로 하여 스트라이핑, 인터리빙, 파이프라이닝과 같은 I/O 병렬 기법들이 연구되었다[1]. 또한, 호스트 인터페이스와 플래시 메모리 간의 직접적인 데이터 전송 경로를 이용하여 데이터 접근 속도를 향상시키는 방안도 제시되었다[2].

이와 함께, 플래시 메모리의 한계를 극복하기 위한 FTL 알고리즘의 연구도 활발하게 진행되었다. FTL의 기본적인 기능은 호스트 인터페이스를 통한 논리적인 주소 체계를 플래시 메모리의 물리적 주소 체계로 변환하는 것이다. 또한 플래시 메모리의 수명의 한계나 in-place update가 불가능한 제약점을 극복하기 위해 wear-leveling과 garbage collection의 기능도 요구된다. SSD의 용량이 증가하면서 주소 체계의 변환을 위한 변환 테이블의 크기가 급격하게 커지면서 기존의 page-mapping 방식을과 block-mapping 방식을 혼합한 hybrid-mapping 알고리즘들이 연구되었다[3][4]. 이들은 기본적으로 작은 크기의 업데이트에 대응하기 위해 page 수준의 주소 변환을 수행함과 동시에 변환 테이블의 크기를 줄이기 위해 block 수준의 주소 변환 역시 수행할 수 있는 구조로 설계되었다.

물론 이 밖에도 시스템 수준에서의 다양한 하드웨어 및 소프트웨어 연구가 수행되었으나, 하드웨어 연구에서는 기본적으로 SSD를 구동하는 마이크로프로세서 및 FTL의 영향을 배제하거나, 반대로 FTL 알고리즘에서는 실제 SSD 하드웨어 상의 제약 조건을 정확하게 고려하지 않아 정확한 성능 예측 및 개선에 활용하기에는 어려움이 존재한다.

2.2 SSD의 내부 구조

본 논문에서는 앞서 언급한 다양한 하드웨어 구조 탐색이 가능하도록 모듈화된 하드웨어 플랫폼을 개발하였다. 개발된 SSD 하드웨어 구조는 그림 1과 같다.

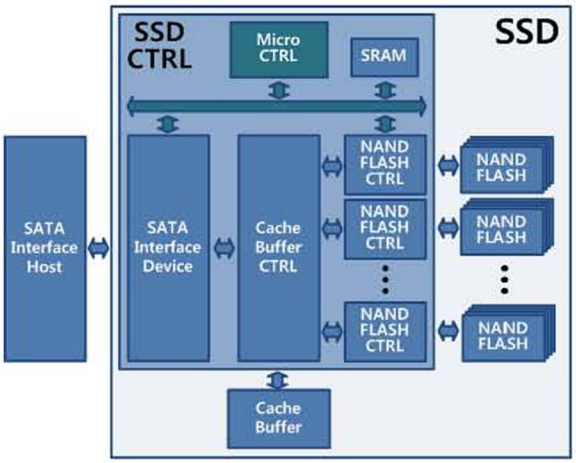

그림 1. SSD 내부 구조

그림 1의 SSD는 크게 다음과 같은 구성 요소들을 가지고 있다.

- 호스트 인터페이스 : 디스크 외부에서의 데이터 접근을 관장하는 구성 요소
- 마이크로컨트롤러 : SSD 전체를 구동하는 FTL을 수행하기 위한 연산 장치
- SRAM 제어기 및 On-chip SRAM: FTL 연산을 수행하면서 필요한 인스트럭션과 데이터를 저장하는 메모리 공간
- 캐시 버퍼 제어기 및 캐시 버퍼: 호스트 인터페이스와 플래시 메모리 사이에서 데이터 읽기 요청 시 캐시에 저장된 과거 데이터를 활용하거나, 데이터 쓰기 요청 시 속도 차이를 극복하기 위한 버퍼 역할을 담당하는 메모리
- 플래시 메모리 제어기 및 플래시 메모리: 각 채널별로 플래시 메모리 장치를 제어하여 실제로 데이터를 읽거나 쓰기 위한 구성 요소

그림 1의 내부 구조는 SystemC 언어를 이용하여 상위 수준에서 모델링되었으므로 각 메모리의 용량 변

경, 마이크로컨트롤러, 채널, 각 메모리의 개수 변경 등 하드웨어 구조 탐색에 용이하다. 또한 기존의 RTL 언어를 이용한 설계보다 더욱 빠른 시뮬레이션이 가능하므로 구조 탐색의 속도가 획기적으로 개선된다.

2.3 FTL 알고리즘

본 논문에서는 BAST 알고리즘을 SSD에서 구동되는 FTL의 알고리즘으로 선정하였다[3]. 물론 본 논문의 시뮬레이션 환경은 특정 알고리즘에 종속되지 않으므로 어떠한 FTL 알고리즘도 실제 하드웨어 플랫폼 상에서 구동하여 정확한 성능을 평가할 수 있다.

BAST 알고리즘은 기본적으로 hybrid-mapping 방식의 주소 체계 변환 알고리즘이다. 따라서 page-mapping이나 block-mapping 방식보다는 복잡한 주소 변환 테이블을 가지게 된다. 또한 이러한 변환 테이블은 SSD 하드웨어의 메모리 계층 구조를 고려하여 접근되어야 한다. 따라서 BAST 알고리즘의 이식 과정에서 이를 고려하여 주소 변환 테이블을 설계하였다. 또한 알고리즘 자체에서 제공하지 않는 wear-leveling 기법과 garbage collection 기법을 함께 구현함으로써 실제 SSD 시스템을 운용할 수 있는 FTL을 탑재하였다. 따라서 BAST 알고리즘 자체에서 필요로 하는 데이터 구조 외에도 다양한 추가의 데이터 구조가 필요하게 된다. 마지막으로, BAST 알고리즘은 플래시 메모리의 채널이나 웨이에 대한 고려가 전혀 없다. 본 논문에서는 BAST 알고리즘을 SSD의 각 채널 및 웨이에 적용하기 위하여 논리적 주소를 추가 할당하고, 인터리빙의 효과를 극대화하기 위해 논리적 주소의 하위 주소를 채널과 웨이에 할당하였다. 이를 통해 각 채널 및 웨이에 독립적으로 BAST 알고리즘을 이식할 수 있게 되었다.

앞서 언급한 다양한 내용을 모두 고려할 경우 하드웨어의 고려 없이 알고리즘만을 수행하는 것과는 달리 정확한 FTL의 실제 성능을 확인할 수 있다.

Ⅲ. 구현

본 논문의 상위 수준 시뮬레이션 환경은 Carbon Design Systems 사의 SoC Designer 프로그램 상에서 개발되었다[6]. 이 프로그램은 SystemC 언어를 이용하여 하드웨어 시스템을 구축하고 이를 소프트웨어와 연동하여 실행할 수 있는 환경을 제공해준다. 이 프로그램 상에서 SSD의 각 구성 요소들을 SystemC로 모델링하고, 목표로 하는 FTL 알고리즘을 이식함으로써 전체 시뮬레이션 환경이 구축되었다.

그림 2와 3은 본 논문의 시뮬레이션 환경을 통해 측정된 SSD 플랫폼의 채널 개수 변화에 따른 성능 변화를 보여준다. 이때 각 채널은 채널 당 4개의 웨이를 갖도록 설정되었으며, 사용된 입력 트래픽은 약 100MB의 용량을 갖는다. 그림 2에서 볼 수 있는 것처럼 채널의 개수가 늘어남에 따라 전체 데이터를 전송하는데 걸리는 시간이 반비례하여 줄어든다. 이는 SSD의 각 채널에서 인터리빙이 효과적으로 이루어지고 있다는 것을 확인해준다. 하지만 2개 채널을 사용한 경우에서 4개 채널을 사용한 경우로 늘어날 경우에는 데이터 전송 시간이 2배로 줄어들지 않는 것을 확인할 수 있다. 이는 채널의 개수가 늘어남에 따라 여러 채널이 동시에 사용되는 빈도가 줄어들면서 인터리빙의 효과가 다소 감소한 것으로 분석할 수 있다.

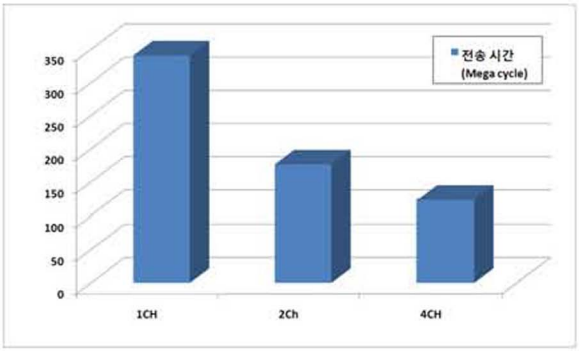

그림 2. 채널 개수 변화에 따른 데이터 전송 시간 비교

그림 3은 이를 Throughput으로 측정하여 나타낸 그림이다. 전체 데이터의 크기가 100MB일 때에는 4개 채널을 이용하여 100MB/s의 속도로 데이터를 전송할 수 있다. 이러한 분석을 통해 설계 시 목표로 하는 전송 속도를 달성하기 위해서는 채널과 같은 하드웨어 자원을 어떻게 배치해야 하는가 등의 탐색에 큰 도움을 받을 수 있다.

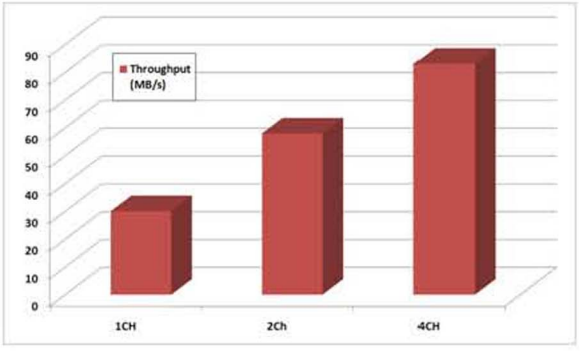

그림 3. 채널 개수 변화에 따른 Throughput 비교

그림 4는 동일한 입력 트래픽과 하드웨어 구조상에서 FTL의 제약 조건을 변경한 실험이다. 이 실험에서 1번의 경우 FTL에 필요한 주소 변환 테이블 전체가 SRAM 위에 존재하여 추가의 데이터가 필요하지 않는 경우이다. 그리고 2번은 SRAM에 주소 변환 테이블의 일부만이 존재하는 경우이다. FTL은 각각의 데이터 요청 시마다 주소 변환 테이블을 필요로 하게 되는데 이 중 일부가 SRAM에 존재하지 않는 경우, 이를 SRAM으로 가져오는데 추가의 수행 시간이 필요로 하게 된다. 그 만큼 데이터의 전송이 지연되므로 그림 4와 같이 Throughput이 저하되는 것을 확인할 수 있다.

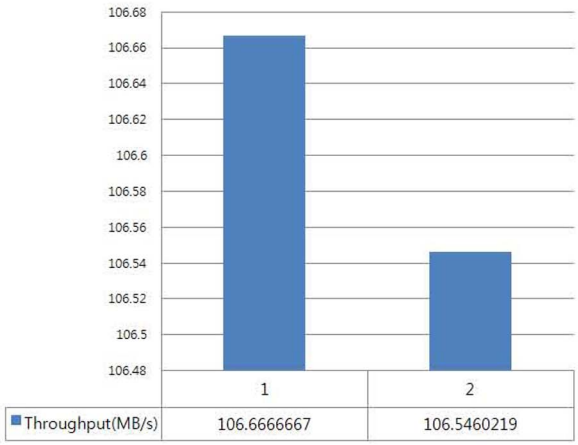

그림 4. FTL의 주소 변환 테이블 크기에 따른 Throughput 변화

Ⅳ. 결론 및 향후 연구 방향

본 논문은 SSD의 성능 향상을 위한 구조 탐색 및 FTL의 성능 평가가 가능한 상위 수준 시뮬레이션 환경을 보여준다. 이 환경에서는 SSD의 각 구성 요소가 다양한 구조로 연결될 수 있으므로 SSD의 다양한 성능 지표의 향상을 위한 빠른 구조 탐색이 가능하다. 또한, 기존의 FTL 알고리즘들이 고려하지 못했던 하드웨어의 제약 조건에 따른 성능 변화를 확인할 수 있다. 마지막으로, 현재의 SSD 구조의 비효율적인 부분을 찾아내어 SSD의 성능을 최적화하는 데에도 유용하게 사용될 수 있다. 향후에는 본 논문의 시뮬레이션 환경을 활용하여 멀티코어를 이용하여 병렬성을 극대화한 SSD 구조와 이때의 병렬 FTL 알고리즘을 연구할 예정이다.

Acknowledgement : 이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원(No. 313-2007-2-D00578), Hynix의 지원 및 IDEC의 지원을 받아 수행된 연구임.

참고문헌

[1] Jeong-Uk Kang 외, "A Multi-channel Architecture for High-performace NAND flash-based Storage System", Journal of Systems Architecture, Vol. 53, pp. 644-685, 2007.

[2] S.-L. Min 외, "Current Trends in Flash Memory Technology", Proceedings of ASP-DAC, pp. 332-333, 2006.

[3] Jesung Kim 외, "A Space-Efficient Flash Translation Layer for CompactFlash Systems", IEEE Transactions on Consumer Electronics, Vol. 48, No. 2, pp. 366-375, 2002.

[4] Sang-Won Lee 외, "A Log Buffer-based Flash Translation Layer using Fully-Associative Sector Translation", ACM Transactions on Embedded Computing Systems, Vol. 6, No. 3, Article 18, 2007.

[5] www.systemc.org

[6] carbondesignsystems.com