LightNVM 지원을 위한 오픈 채널 SSD 에뮬레이터 구현

진주영○, 신동군
성균관대학교 전자전기컴퓨터공학과
siennajjy@skku.edu, dongkun@skku.edu

Implementation of Open-channel SSD Emulator supporting LightNVM

Jhuyeong Jhin○, Dongkun Shin
Department of ECE, Sungkyunkwan University

요 약

솔리드 스테이트 드라이브(SSD) 내부에는 FTL(Flash Translation Layer)이라는 소프트웨어 레이어가 내장된다. FTL 알고리즘이 SSD의 성능에 큰 영향을 끼치기 때문에 이를 최적화하는 연구들이 활발하게 진행되어 왔다. 그러나 FTL은 디바이스에 내장되어 있기 때문에 제한된 정보만을 활용할 수 있다. 이러한 배경에서 오픈 채널 SSD(Open-channel SSD)가 제안되었다. 오픈 채널 SSD는 SSD 디바이스로, LightNVM을 통하여 입출력 요청을 받는다. LightNVM은 호스트 사이드 드라이버로, 입출력 요청과 관련한 다양한 의미 정보를 활용하여 기존의 FTL이 수행하던 기능들의 일부를 대신 수행할 수 있다. 그러나 현재는 공개된 오픈 채널 SSD 디바이스는 고가이고, 관련 정보가 적기 때문에 오픈 채널 SSD를 연구하는데 어려움이 있다. 본 논문에서는 오픈 채널 SSD 연구를 위해 커널 모듈 형태의 에뮬레이터를 구현하였다. 제안하는 에뮬레이터는 LightNVM을 통해 입출력 요청 받아 휘발성 메모리에 입출력 동작을 수행하는 가상의 디바이스이다. 오픈 채널 SSD 에뮬레이터를 이용하면, 물리적인 오픈 채널 SSD 장치 없이 LightNVM을 통해 입출력 요청을 받을 수 있다. 따라서 오픈 채널 SSD에 관한 연구를 더욱 용이하게 한다.

1. 서 론

플래시 메모리는 저전력, 고성능, 높은 내구성의 장점을 가진다. 최근 플래시 메모리의 가격이 빠르게 하락하고, 용량이 커지고 있다. 따라서 낸드 플래시 메모리 기반의 저장장치인 솔리드 스테이트 드라이브(SSD)가 기존의 하드디스크 드라이버(HDD)를 대체하고 있다. SSD는 여러 개의 플래시 칩, DRAM 버퍼와 컨트롤러로 구성되며, 멀티 채널과 멀티 웨이 구조를 활용하여 여러 칩에 동시에 읽기/쓰기 연산이 가능하다.

플래시 메모리 칩은 여러 개의 블록으로 구성되어 있고, 각 블록은 여러 개의 페이지를 가진다. 페이지는 읽기/쓰기 연산의 단위이며, 페이지에 데이터를 기록하기 전에 반드시 해당 블록을 지워야 하는 특성을 가진다. 그러나 지우기 연산은 블록 단위로 이루어지므로, 쓰기 연산을 하고자 하는 페이지와 같은 블록에 속한 다른 페이지들을 복사해야 한다. 이러한 특성들을 다루기 위해, SSD 내부에는 FTL(Flash Translation Layer)이라는 소프트웨어 레이어가 내장된다.

FTL은 플래시 메모리의 성능 향상과 수명 연장을 위해 효율적인 L2P 주소 매핑 방식을 제공하고 가비지 컬렉션(GC; Garbage Collection), 웨어-레벨링(Wear-leveling)과 같은 여러 가지 테크닉을 사용한다. 따라서 FTL의 동작 알고리즘은 SSD의 성능에 크게 영향을 미친다. 이러한 FTL의 중요성이 부각되면서 FTL 알고리즘을 최적화시키기 위한 연구들이 진행되어 왔다. 그러나 저장장치에 내장되어 있는 FTL이 입출력 요청과 관련하여 가지는 정보는 매우 제한적이므로 성능 향상을 이루기에는 한계가 있다.

위와 같은 배경을 바탕으로 오픈 채널 SSD(Open-channel SSD)[1]가 제안되었다. 오픈 채널 SSD는 물리적인 저장공간을 운영체제에 드러내고 커널 레벨에서 이를 관리하도록 하는 SSD 디바이스이다. 따라서 호스트 시스템은 기존의 SSD 장치가 내장된 FTL을 이용하여 제한된 정보만을 가지고 수행해온 동작들의 일부를 수행한다. 오픈 채널 SSD는 LightNVM[2]을 통해 입출력 요청을 받는다. LightNVM은 커널 모듈 형태로 구현된 호스트 사이드 드라이버로, 커널 레벨에서 SSD 저장공간을 관리하고 기존 FTL 기능의 일부를 수행한다. LightNVM의 소스 코드는 리눅스 커널 4.4 버전부터 커널 메인 라인에 포함되었다.

Memblaze, PMC Sierra, Micron 등 여러 벤더들이 오픈 채널 SSD 디바이스를 개발 중이지만 공개된 오픈 채널 SSD 디바이스의 수가 적고, 고가이며 관련 정보가 적기 때문에 오픈 채널 SSD와 관련된 연구를 진행하기에 어려움이 있다.

본 논문에서는 오픈 채널 SSD 연구를 위해 커널 모듈 형태의 에뮬레이터를 구현하였다. 제안하는 에뮬레이터는 휘발성 메모리에 입출력 동작을 수행하는 가상의 디바이스를 전제하고, LightNVM을 통해 입출력 요청을 받아 처리한다. 오픈 채널 SSD 에뮬레이터를 이용하면, 물리적인 오픈 채널 SSD 장치 없이 LightNVM을 통해

이 논문은 2016년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2016R1A2B2008672)

입출력 요청을 받을 수 있다. 따라서 입출력 요청 처리 과정과 오버헤드를 관찰할 수 있다. 또한 현재 구현되어 있는 Target에 대한 연구나, 새로운 Target 구현에 대한 연구도 진행할 수 있다. 에뮬레이터의 정상적인 동작을 확인하기 위해서 fio를 이용하여 임의 쓰기와 임의 읽기 동작을 수행해보았다.

2. 관련 연구

오픈 채널 SSD는 LightNVM을 통해 입출력 요청을 받는다. LightNVM은 커널 모듈 형태의 호스트 사이드 드라이버로, L2P 주소 매핑 테이블 관리와 가비지 컬렉션 같이 기존의 FTL 동작들의 일부분을 수행한다. LightNVM의 소스 코드는 리눅스 커널 4.4 버전부터 커널 메인 라인에 포함되었다.

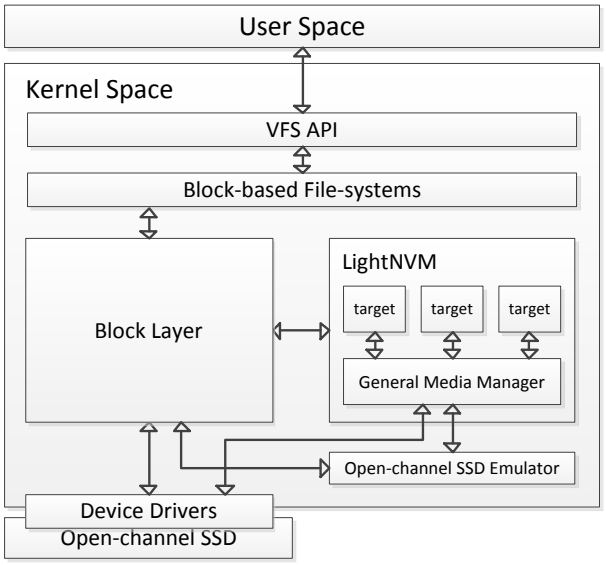

그림 1 LightNVM과 커널 소프트웨어 레이어

그림 1은 LightNVM의 간략한 내부 구조와 커널 소프트웨어 레이어들 사이의 관계를 대략적으로 나타낸다. LightNVM은 크게 Target과 General Media Manager로 구성된다.

Target은 주로 L2P 매핑 관리와 가비지 컬렉션 동작을 수행한다. 각각의 낸드 플래시 칩들은 서로 다른 Target을 이용할 수 있다. 따라서 칩 별로 L2P 주소 매핑 방식과 가비지 컬렉션 정책을 선택할 수 있다. 커널은 Target과 연결된 각 칩들을 독립적인 디바이스로 간주한다. 현재 pblk, DFlash와 같은 여러 Target들이 구현 중에 있다. 그러나 현재 리눅스 커널 소스코드의 메인라인에 포함되어있는 것은 RRPC(Round-robin Page-based Hybrid FTL) 뿐이다. RRPC Target은 리눅스 커널 4.4버전부터 메인라인에 포함된 LightNVM의 유일한 Target이다. RRPC Target의 L2P 주소 매핑 관리는 선형 주소를 단순히 Round-robin 방식, 즉 0부터 1씩 순차적으로 증가시키며 부여하는 정책을 채택하고 있다.

General Media Manager는 오픈 채널 SSD 디바이스의 등록 과정에서, 디바이스 드라이버로부터 물리적인 저장장치의 정보를 전달 받는다. 전달 받은 저장장치의 정보를 이용하여 디바이스를 추상화하는 객체를 생성하고 디바이스를 관리하는 정보로 활용한다. 또한 General Media Manager는 Target들과 낸드 플래시 칩들 간의 연결 정보와 통신을 관리한다.

오픈 채널 SSD의 디바이스 드라이버는 LightNVM에 등록하는 과정을 거쳐 LightNVM으로부터 입출력 연산을 받을 수 있게 된다. 이 과정에서 디바이스 드라이버는 해당 디바이스의 물리적인 저장장치의 정보, 즉 낸드 플래시 칩, 채널, 플레인, 블록 개수 등을 General Media Manager로 전달한다. 또한 LightNVM에 등록된 장치는 낸드 플래시 칩 별로 어느 Target을 이용할 것인지 명시하여 특정 Target과 연결되어야 한다. 오픈 채널 SSD는 각 칩들과 Target의 연결 과정까지 마치면 LightNVM을 통해 입출력 요청을 받을 수 있다.

파일 시스템들이 저장장치로 입출력 요청을 보내면, 해당 요청은 입출력이 수행될 위치와 데이터의 포인터, 크기를 포함하는 bio 객체 형태로 변환되어 블록 레이어로 보내진다. LightNVM은 블록 레이어로부터 입출력 요청을 받는다. Target이 L2P 주소 변환과 가비지 컬렉션, 웨어-레벨링 등 기존 FTL 기능의 일부를 수행하고, General Media Manager가 다시 입출력 요청을 디바이스 드라이버로 보낸다.

Memblaze, PMC Sierra, Micron 등 여러 벤더들이 오픈 채널 SSD 디바이스를 개발 중이다. 그러나 공개된 오픈 채널 SSD 디바이스의 수가 적고, 고가이며, 관련 정보가 부족하기 때문에 오픈 채널 SSD와 관련된 연구를 진행하기에 어려움이 있다.

리눅스 커널 메인라인에는 LightNVM을 통해 입출력 요청을 받을 수 있는 Null Block 디바이스 드라이버가 포함되어 있다. 따라서 Null Block 디바이스 드라이버를 이용하면, LightNVM을 통한 입출력 요청을 관찰할 수 있다. 또한 현재 구현되어 있는 Target에 대한 연구나, 새로운 Target 구현에 대한 연구도 진행할 수 있다. 그러나 Null Block 디바이스 드라이버는 단순히 LightNVM로부터 입출력 요청을 받을 수만 있을 뿐, 실제로 입출력 연산을 수행하지는 않는다. 따라서 실제 입출력 연산 수행 결과를 필요로 하는 어플리케이션 동작, 예를 들면 SQLite의 WAL 방식과 같은 동작을 실험하는 것은 불가능하다.

3. 오픈채널 SSD 에뮬레이터

기존의 SSD 장치에 내장된 FTL의 한계성이 드러나면서, 호스트 시스템이 보다 다양한 의미 정보를 활용하여 FTL의 기능을 일부 수행하는 오픈 채널 SSD가 제안되었다. 본 논문에서는 LightNVM을 통해 입출력 요청을 받아 메모리에 입출력 연산을 수행하는 오픈 채널 SSD 에뮬레이터를 구현하였다. 오픈 채널 SSD 에뮬레이터를 이용하면 물리적인 오픈 채널 SSD 장치 없이 LightNVM을 통해 입출력 요청을 받아 관련 연구를 진행할 수 있다.

그림 1에서 구현된 오픈 채널 SSD 에뮬레이터가 커널 레벨에 속함과, LightNVM과 통신함을 표현하였다. 오픈 채널 SSD 에뮬레이터는 모듈 형태로 구현되었다. 실제 입출력 연산은 메모리에 수행함으로써 가상의 디바이스를 전제로 하기 때문에, 실제 물리적인 비휘발성 저장장치와 통신하지 않는다.

에뮬레이터는 블록 디바이스 드라이버 등록 후, LightNVM에 등록하는 과정을 거쳐야만 LightNVM을 통해 입출력 요청을 받을 수 있다. 입출력 요청이 발생하였을 때, 블록 레이어가 bio 객체 형태로 전달 받는다. 블록 레이어는 이를 디바이스와 연결된 Target에게 전달한다. Target은 L2P 주소 변환을 수행한 후, LightNVM의 입출력 요청 객체에 관련 정보와 요청 동작을 식별하는 opcode를 초기화시키고 General Media Manager에게 전달한다.

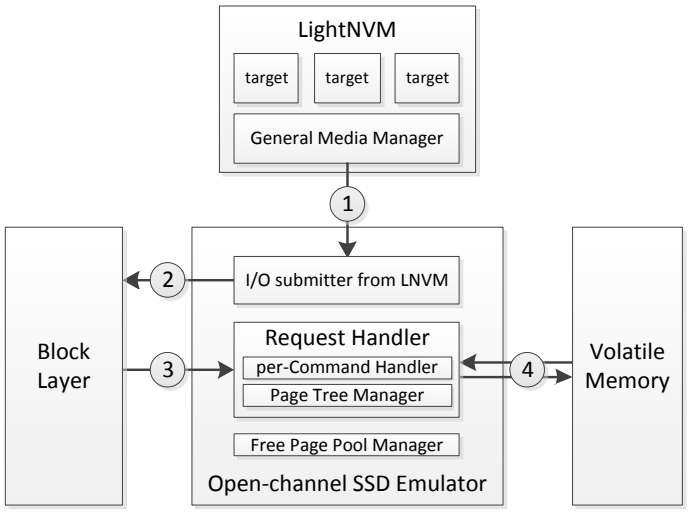

그림 2 오픈 채널 SSD 에뮬레이터 입출력 처리 과정

그림 2는 오픈 채널 SSD 에뮬레이터가 LightNVM으로부터 전달 받은 입출력 요청을 처리하는 과정을 추상화하였다. : 1) General Media Manager는 Target으로부터 초기화된 요청 정보를 전달 받아 오픈 채널 SSD 에뮬레이터의 I/O submitter로 전달한다. 이 요청 정보는 LightNVM의 입출력 동작을 완료하는 함수의 포인터와 bio 객체, 요청 큐 포인터와 요청 동작을 식별할 opcode 등을 포함한다. I/O submitter는 request 구조체의 객체를 할당 받아, 전달 받은 요청 정보를 이용하여 초기화한다. 2) 초기화된 request 구조체의 객체와 요청 큐 등을 Block Layer로 보낸다. Block Layer는 전달 받은 request 구조체 객체를 요청 큐에 삽입하고, 큐를 동작시키는 함수를 호출한다. 3) Block Layer는 다시 오픈 채널 SSD 에뮬레이터의 Request Handler로 요청 큐를 전달한다. 오픈 채널 SSD 에뮬레이터의 Request Handler는 요청 큐로부터 처리할 request 구조체 객체를 꺼내서 opcode를 식별하여 알맞은 동작을 수행한다. 4) 이 때, 읽기와 쓰기 동작은 메모리에 수행하게 된다. 입출력 요청에 쓰이는 메모리 공간은 미리 할당해 놓지 않고 필요할 때마다 페이지를 할당 받는다. 이렇게 할당 받은 페이지들은 라딕스 트리로 관리하였다. 할당 받은 페이지의 해지는 에뮬레이터 모듈이 unload 될 때를 제외하면 ERASE 요청을 수행할 때만 발생한다. 페이지 할당/해지의 반복이 빈번하게 발생하는 것을 방지하기 위하여, ERASE 요청 처리 시 페이지를 바로 해지하지 않고 연결 리스트로 구현된 별도의 메모리 풀에 넣어 관리한다.

4. 실 험

본 논문에서는 오픈 채널 SSD 에뮬레이터가 실제로 LightNVM을 통해 입출력 요청을 받아 처리하는 지 검증하였다. LightNVM으로부터 입출력 요청을 받기 위하여 오픈 채널 SSD 에뮬레이터를 LightNVM에 등록하고, RRPC Target과 연결하였다.

표 1 워크로드 별 I/O 요청 횟수와 Command 처리 횟수

	R/W	R/R	S/W	S/R
I/O 요청 횟수	25600	25600	25600	25600
Command 처리 횟수	25600	25600	25600	25600

표 1은 fio를 이용하여 4K 씩 총 100MB의 데이터 임의쓰기, 임의읽기, 순차쓰기, 순차읽기를 발생시킨 후, 어플리케이션이 요청하는 입출력 동작만큼 디바이스 커맨드가 발생하는 지 확인한 결과이다. 디바이스 커맨드 발생 및 처리 횟수는 Generic Media Manager가 호출하는 오픈 채널 SSD 에뮬레이터의 I/O Submitter 함수에서 읽기와 쓰기를 구별하여 누적하였다. 표에서 R/W, R/R, S/W, S/R는 각각 순서대로 임의쓰기, 임의읽기, 순차쓰기, 순차읽기를 의미한다. 네 가지 워크로드에서 모두 I/O 요청 횟수와 에뮬레이터에서의 커맨드 발생 및 처리 횟수가 일치하였다. 이를 통해 에뮬레이터가 LightNVM으로부터 정상적으로 입출력 요청을 받아 처리함을 알 수 있다.

5. 결론 및 향후 연구

FTL 최적화 연구의 한계를 극복하기 위해 오픈 채널 SSD가 제안되었으나, 공개된 오픈 채널 SSD 디바이스는 고가이고, 관련 정보가 적기 때문에 연구하는데 어려움이 있다. 본 논문에서는 오픈 채널 SSD 연구를 위해 커널 모듈 형태의 에뮬레이터를 구현하였다. 제안하는 에뮬레이터는 LightNVM을 통해 입출력 요청을 받아 메모리 공간에 해당 요청 동작들을 수행한다. 물리적인 오픈 채널 SSD 장치 없이도 오픈 채널 SSD에 관한 연구를 더욱 용이하게 한다.

오픈 채널 SSD 에뮬레이터를 이용하여, 새로운 Target 구현에 대한 연구를 진행할 계획이다. 특히 커널 레벨에서는 입출력 요청 데이터의 성격, 즉 메타데이터인지, 저널 데이터인지 등을 알 수 있으므로 이를 이용하여 효과적인 Data Placement 알고리즘을 연구할 수 있을 것이다.

참 고 문 헌

[1] Bjørling, Matias, et al. "Linux Kernel Abstractions for Open-Channel Solid State Drives." Non-Volatile Memories Workshop. 2015.

[2] Bjørling, Matias, et al. "LightNVM: Lightning Fast Evaluation Platform for Non-Volatile Memories." Non-Volatile Memories Workshop. 2014.