ZNS를 이용한

파일시스템 성능 개선 연구

Team: OS 을~매나 맛있게요?

부산대학교 전기컴퓨터공학부 정보컴퓨터공학전공

School of Electrical and Computer Engineering, Computer Engineering Major

Pusan National University

2022년 5월 16일

지도교수: 안 성 용 (인)

목차

1. 과제 배경 및 목표

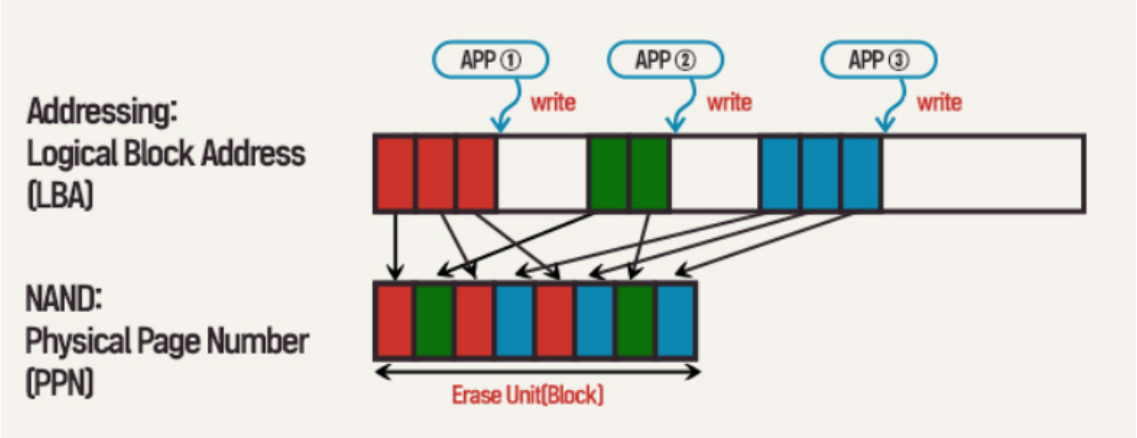
1. ZNS(Zone Namespace 배경)

최근 각광받고 있는 4차 산업의 핵심기술을 ABCD라고 이야기한다. ABC는 각각 인공지능(AI), 빅데이터(Big Data), 클라우드(Cloud)를 말한다. 그리고 이 기술들은 모두 'D', 데이터센터(Datacenter)를 통해 동작한다. 때문에 세계 유수 기업들은 데이터센터의 성능과 비용효율을 높이기 위해 꾸준히 연구해오고 있다.

과거 데이터센터는 하나의 성능 좋은 컴퓨터(Server)에서 하나의 운영체제와 응용프로그램을 구동했다. 반면 차세대 데이터센터는 하드웨어 풀(Pool)위에 다수의 가상 운영체제와 응용프로그램이 구동되는 형태이다. 즉, 다양한 종류의 프로그램이 하나의 저장장치를 동시에 읽고 쓰는 것이다. 따라서 데이터센터에서는 속도뿐 아니라 다중 사용 환경(multi-tenant)에서도 서로 간섭없이 일정한 성능을 제공하는 서비스 품질(Quality of Services, QoS)을 갖춘 저장 장치가 요구된다.

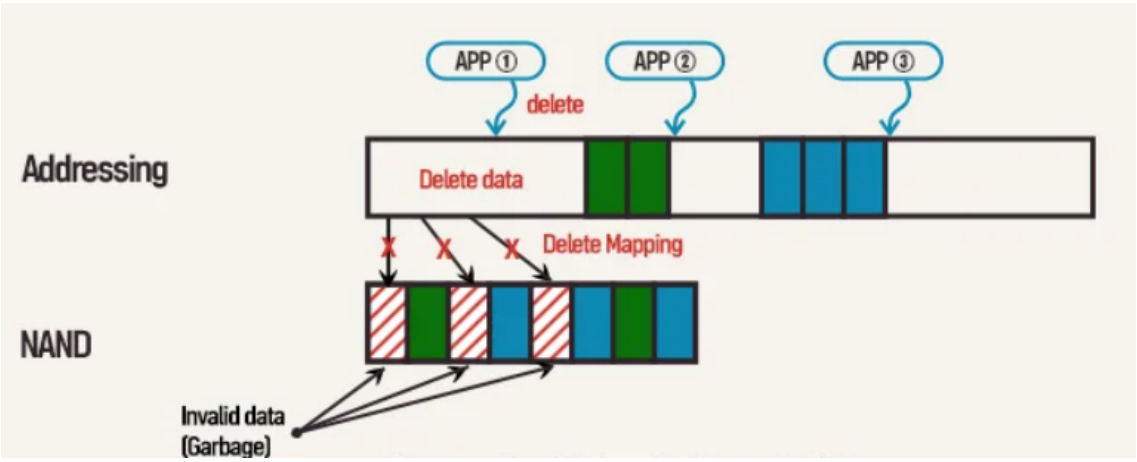
SSD는 HDD(하드디스크)에 비해 굉장히 빠른 성능을 가진 저장 장치이다. 그러나 현재 flash-based SSD는 수 십 년된 block interface를 가지고 있다. 이로 인해 capacity over-provisioning, page mapping tables을 위한 DRAM(Dynamic Random-Access Memory), Garbage Collection Overheads, 그리고 Garbage collection을 완화하기 위해 쓰이는 호스트 소프트웨어의 복잡성 측면에서 상당한 비용이 발생한다. [1]

이후 해결책으로 제안된 것이 차세대 기업용 ZNS SSD이다. ZNS는 Zoned Namespace의 약자로, Namespace를 Zone 단위로 나눠 사용하는 기술을 말한다.



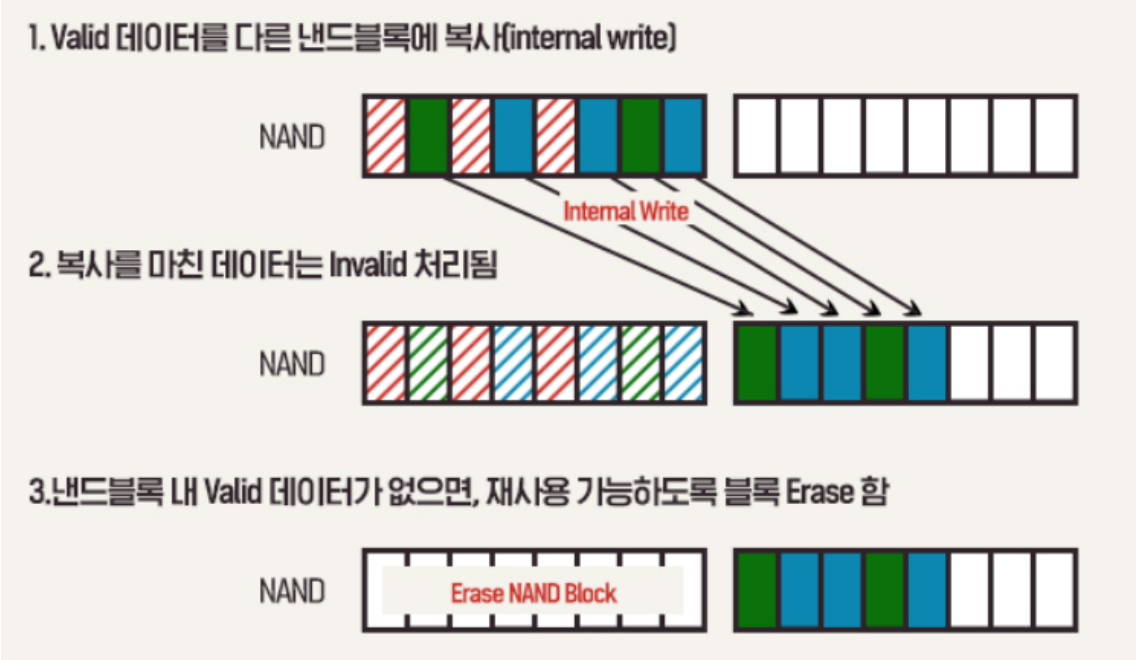
[그림 1. Conventional SSD에서의 Write data process]

응용 프로그램(APP)이 SSD의 각자의 영역(LBA)에 데이터를 저장(Write)하면, SSD는 하나의 낸드블록에 순차적으로 데이터를 저장하고 데이터가 저장된 물리주소(PPN)과 논리주소(LBA)를 mapping한다.



[그림 2. SSD에서의 Delete data process]

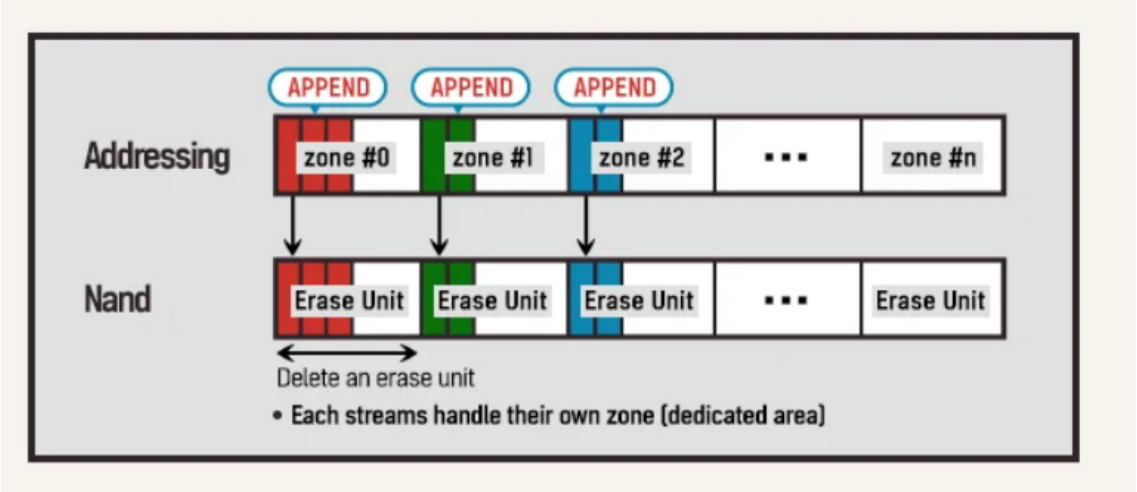
APP에서 SSD의 데이터를 삭제하면, SSD는 논리주소(LBA)와 물리주소(PPN)의 mapping만 삭제한다. NAND에는 기존 데이터가 가비지(Garbage, 즉, Invalid data)로 남아 있게 된다.



[그림 3. SSD에서의 Garbage Collection process]

Write와 Delete Data를 반복하면서 NAND Block이 가득 채워지면 Balid data만 모아 다른 NAND Block에 복사를 한다. 복사를 마친 기존의 Balid data는 Invalid data로 처리되고 NAND Block 내에 Valid 데이터가 없으면, 재사용 가능하도록 블록 Erase를 한다. 이 과정을 Garbage Collection이라고 하는데, 이것이 발생하면 진행 중이던 Read/Write process가 멈추기 때문에 서비스 품질(QoS)이 떨어지게 된다.

이 것을 포함한 여러 단점을 보완하기 위해 2021년 6월 2일, 삼성전자는 ZNS(Zoned Namespace) 기술을 적용한 차세대 기업 서버용 Solid State Drive(ZNS SSD)를 출시하였다. ZNS는 SSD 전체 저장 공간을 작고 일정한 용량의 Zone으로 나누구, 용도 및 사용 주기가 같은 데이터를 동일 구역에 저장해 SSD를 효율적으로 활용할 수 있게 만드는 차세대 기술이다.



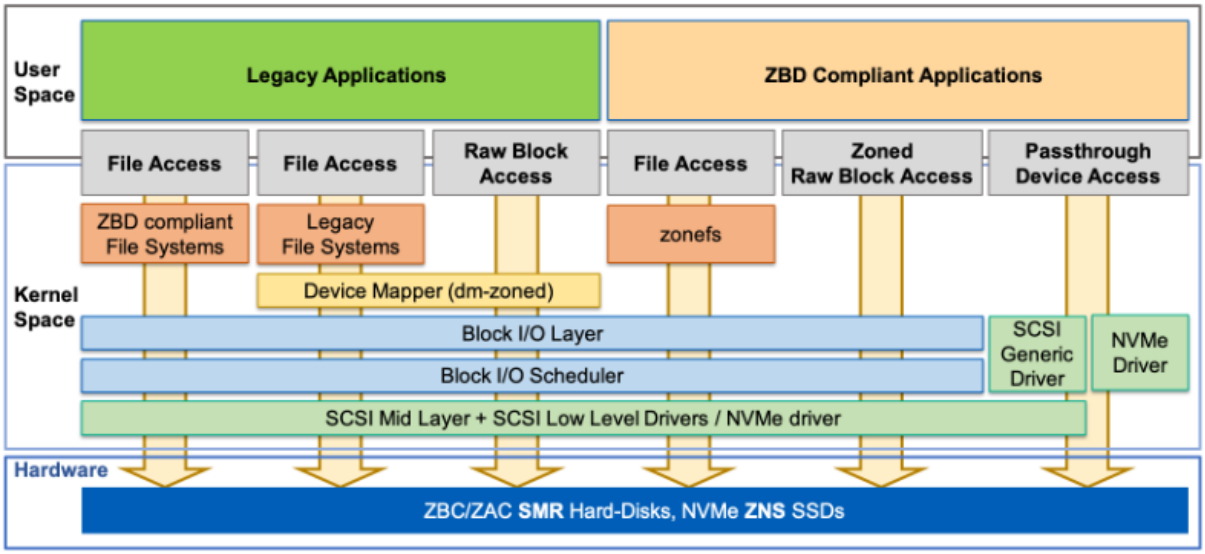
[그림 4. ZNS에서의 Write Data Process]

기존 SSD에서는 여러 응용프로그램이 논리적으로 각자 원하는 영역에 데이터를 저장하더라도, 실제로는 하나의 NAND 블록에 순차적으로 데이터가 저장된다.

반면, ZNS SSD에서는 그림 4와 같이 여러 응용 프로그램이 각자 정해진 Zone에 순차적으로 데이터를 저장한다. 여기서 Zone은 논리적인 공간처럼 NAND 블록에도 나누어져 있다. 하나의 Zone 안에는 비슷한 데이터끼리 모여 있으며, 순차적으로 저장했다가 존 단위로 지우기 때문에 Garbage가 발생하지 않는다. 따라서 SSD에서의 Garbage Collection이 없어도 됨으로써 발생했던 여러 오버헤드를 제거할 수 있다.

2. ext4 filesystem 을 선택한 계기

본 연구의 목적은 ZNS 를 이용하여 파일 시스템의 성능을 개선하고자 하는 것이다.



[그림 5. Linux Zoned block device support overview]

그림 5 에서 File Access 와 Raw Block Access I/O 경로를 통해 legacy applications (완전한 fully-sequential write stream 을 구현하도록 수정되지 않은 application)을 실행할 수 있다. [3] 그 중 File Access interface 는 응용 프로그램이 데이터를 파일 및 디렉토리로 구성할 수 있도록 파일 시스템이 구현하는 인터페이스이다. File Access interface 의 두가지 다른 구현을 사용할 수 있는 데 ZBD Compliant File Systems 과 Legacy File System 이 있다.

ZBD Compliant File System 은 이 구현을 통해 파일 시스템은 File System 을 직접 수정하여 Application 의 파일에 대한 Random write 는 File System 에 의해 sequential write stream 으로 변환되어 Application 에서 device constraints(장치 제약 조건)을 숨긴다. 그 예로 수정된 F2FS(ZNS) file system 이 있다. [2]

Legacy File System 은 수정되지 않은 file system 이 사용되며 device-sequential write constraint(장치 순차 쓰기 제약 조건)은 zoned block device 를 regular block device 로 변환하여 device mapper target driver 에 의해 처리된다. 이 device mapper 를 dm-zoned 라고 한다. 즉, dm-zoned device mapper target 은 sequential write 만 가능한 zoned block devices 에 random write access 를 제공한다.

본래 ext4 file system 은 host-managed zone block device(호스트 관리 장치 제약 조건)에 대한 지원을 제공하지 않는다. 그 이유는 ext4 설계의 기본적인 측면 중 일부는 host-managed device constraint 와 일치하도록 쉽게 변경할 수 없기 때문이다. 그러나 dm-zoned device mapper target 를 사용하므로써 ext4 file system 의 개선 연구가 가능하게 되었다.

저자는 dm-zoned device mapper target 을 이용하여 ext4 file system 의 성능 개선을 연구할 계획에 있다.

2. 과제 세부사항

1. dm-zoned 에 대한 상세 설명

2. ext4 파일 시스템에 대한 상세 설명

4. 개발 일정 및 역할 분담

4.1 현재 진행 상황

<table>
<tr><th colspan="5">6월</th><th colspan="4">7월</th></tr>
<tr><td>2주</td><td>3주</td><td>4주</td><td>5주</td><td>1주</td><td>2주</td><td>3주</td><td>4주</td><td>5주</td></tr>
<tr><td colspan="4">리눅스 커널 스터디</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td colspan="3">ZNS 논문 분석 및 스터디</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td colspan="3">ZNS+ 논문 분석 및 스터디</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="2">Dm-zoned, ext4 스터디 및 femu에서 mount</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="2">filesystem 성능 테스트</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="2">중간보고서 작성</td></tr>
</table>

4.2 향후 개발 일정

<table>
<tr><th colspan="5">8월</th><th colspan="4">9월</th></tr>
<tr><td>1주</td><td>2주</td><td>3주</td><td>4주</td><td>5주</td><td>1주</td><td>2주</td><td>3주</td><td>4주</td><td>5주</td></tr>
<tr><td colspan="4">Ext4 파일 시스템 수정 및 개선</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

			Ftrace를 통한 함수 추적					
				디버깅				
					안정성, 성능 평가			
						디버깅 및 설계문서 수정		
							최종보고서 작성, 발표심사준비	

4.3 역할 분담

이름	역할 분담
임경민	- 보고서, 문서 자료 수집 및 작성 - Emulated ZNS SSD 작업환경 구축 - dm-zoned device mapper 와 ext4 파일 시스템 mount - fio를 통한 성능 측정 - 논문 분석
이선후	
김상현	

참고 문헌

[1] 손한근 TL, “[궁금한 반도체 WHY] ZNS SSD, 기존 SSD와 무엇이 다를까?” 04 18 2019

[2] Matias Bjørling, Western Digital; Abutalib Aghayev, The Pennsylvania State University; Hans Holmberg, Aravind Ramesh, and Damien Le Moal, Western Digital; Gregory R. Ganger and George Amvrosiadis, Carnegie Mellon University, “ZNS: Avoiding the Block Interface Tax for Flash-based SSDs” [온라인]. ZNS: Avoiding the Block Interface Tax for Flash-based SSDs | USENIX

[3] Zoned Storage 사이트의 Linux Zoned Storage Support Overview [온라인]. Linux Kernel Zoned Storage Support | Zoned Storage

[4]