

# Non Volatile Memory (NVM)

- Computer Memory Hierarchy
- Types of Storage Device
- Why SSD is favorite (HDD vs SSD)
- Explain SSD
  - Composition
  - Basic Operation
  - Write Amplification
  - Wear Leveling
  - Flash Translation Layer
  - Internal Parallel Processing
- System Optimization
  - 3D Xpoint
    - Composition
    - Pros.
  - 3D Xpoint vs NAND Flash
  - Summarize SSD

# Computer Memory Hierarchy

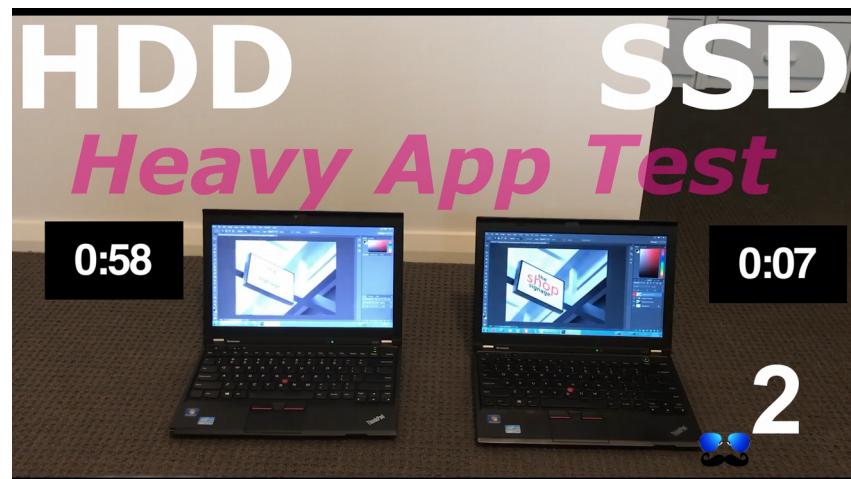
LEVEL	ACCESS TIME	TYPICAL SIZE
Registers	"instantaneous"	under 1KB
Level 1 Cache	1-3 ns	64KB per core
Level 2 Cache	3-10 ns	256KB per core
Level 3 Cache	10-20 ns	2-20 MB per chip
Main Memory	30-60 ns	4-32 GB per system
Hard Disk	3,000,000-10,000,000 ns	over 1TB

## SRAM      vs    DRAM

SRAM : Static Random Access Memory, 컴퓨터의 L1 L2 L3 cache에 주로 사용  
한번 데이터를 입력하면 수정하기 전까지 상태가 변화하지 않음

DRAM : Dynamic Random Access Memory, 컴퓨터의 메인 메모리에 주로 사용  
데이터 상태를 유지하기 위해서 주기적인 전원 충전 (refresh)이 필요

# HDD vs SSD



70

65

60

55

50

45

40

35

30

25

20

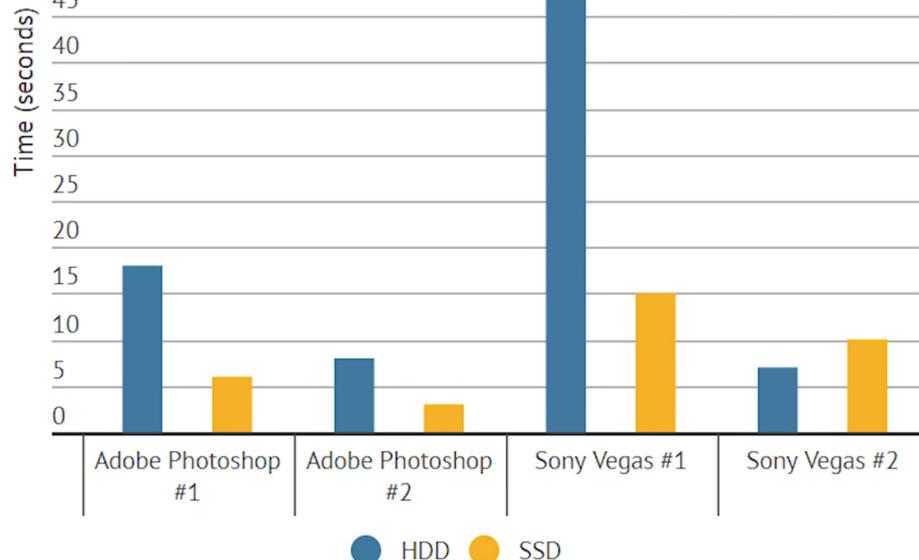
15

10

5

0

# HDD vs SSD



30

25

20

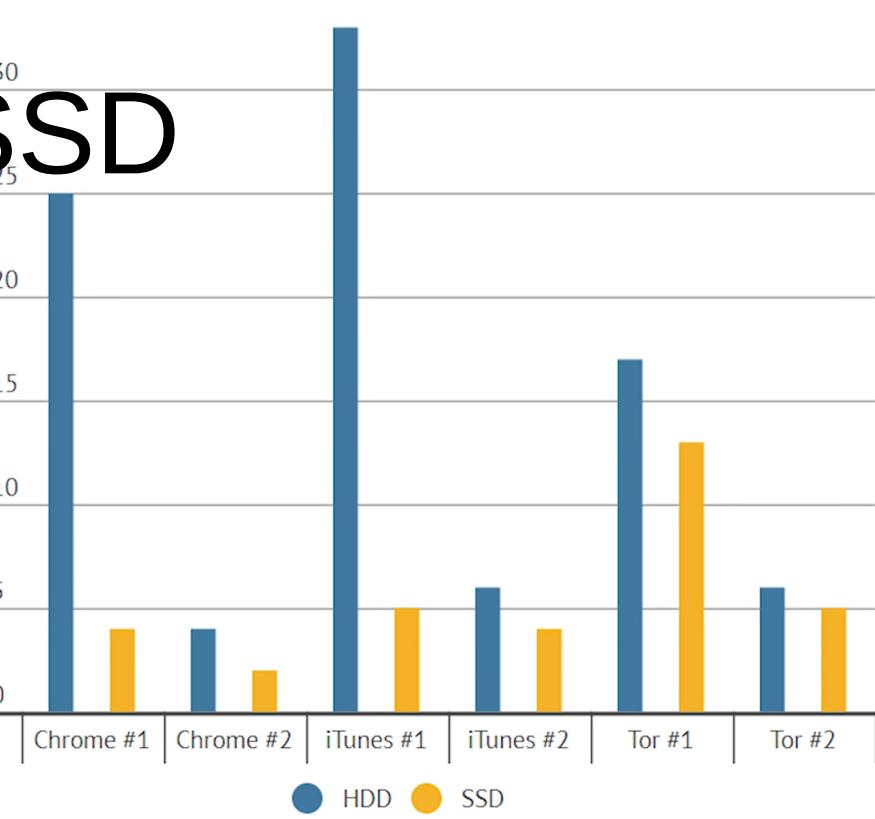
15

10

5

0

Time (seconds)



240

220

200

180

160

140

120

100

80

60

40

20

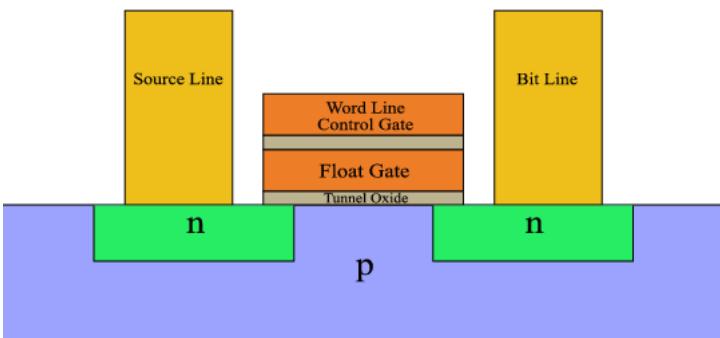
0

Time (seconds)



# NAND Flash Memory

- Limited P/E number  
(Program & Erase)



Control Gate 에는 +  
Floating Gate 에는 -  
fg 의 전하량으로 데이터 기록

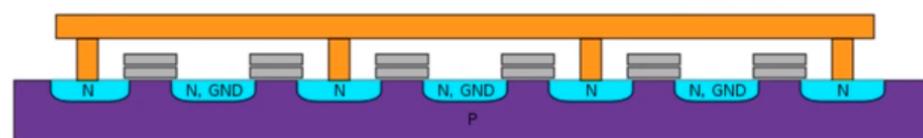
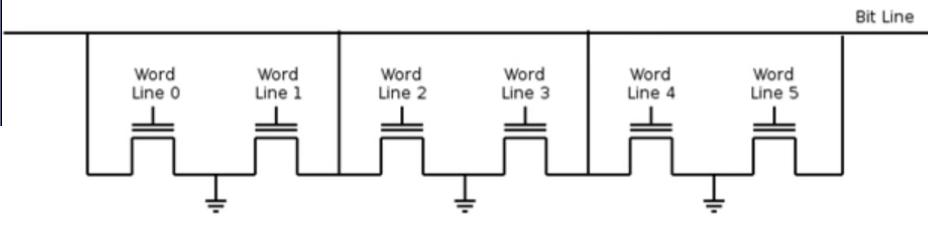
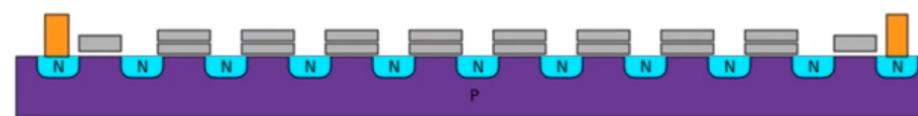
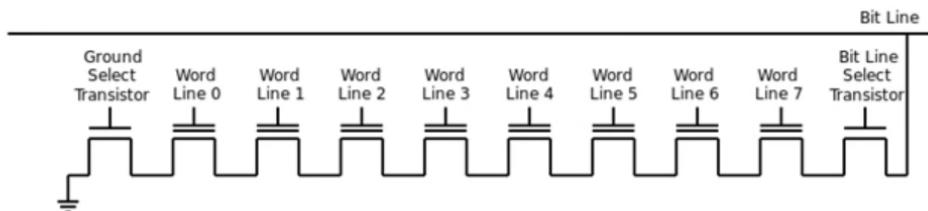
Initial State 는 Bit = 1 이다 .

Read 할 때는 관계없는 셀의 word line 에 강한 신호 Yes, control gate +, floating gate 의 - 전하를 불잡아둠

읽으려는 셀 word line 에는 약한 신호 Yes, floating gate 의 데이터가 흘러가도록 만듬

Write 할 때는 연쇄적인 모든 셀의 word line 에 신호 Yes,  
쓰려는 셀 word line 에는 강한 신호 Yes,

기록한 데이터를 erase 하려고 할 때마다 약간의 - 전하가 oxide layer 안에 남아서 축적  
제한된 P/E Cycle 일 수 밖에 없다



## NAND

- Higher density
- Used for:
  - USB drives
  - memory cards
  - solid state drives

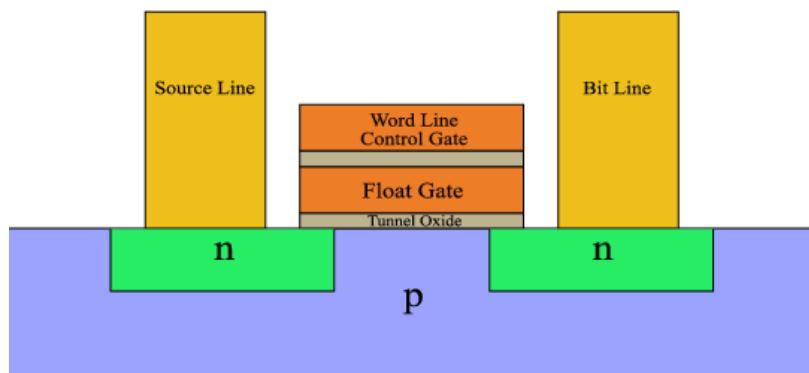
## NOR

- Lower latency
- Used for embedded systems

# NAND Flash Memory

- SLC, MLC, TLC

Floating Gate에 포함된 – 전하량 인식 가능  
여러 단계로 나눠서 사용하는 것



수명은

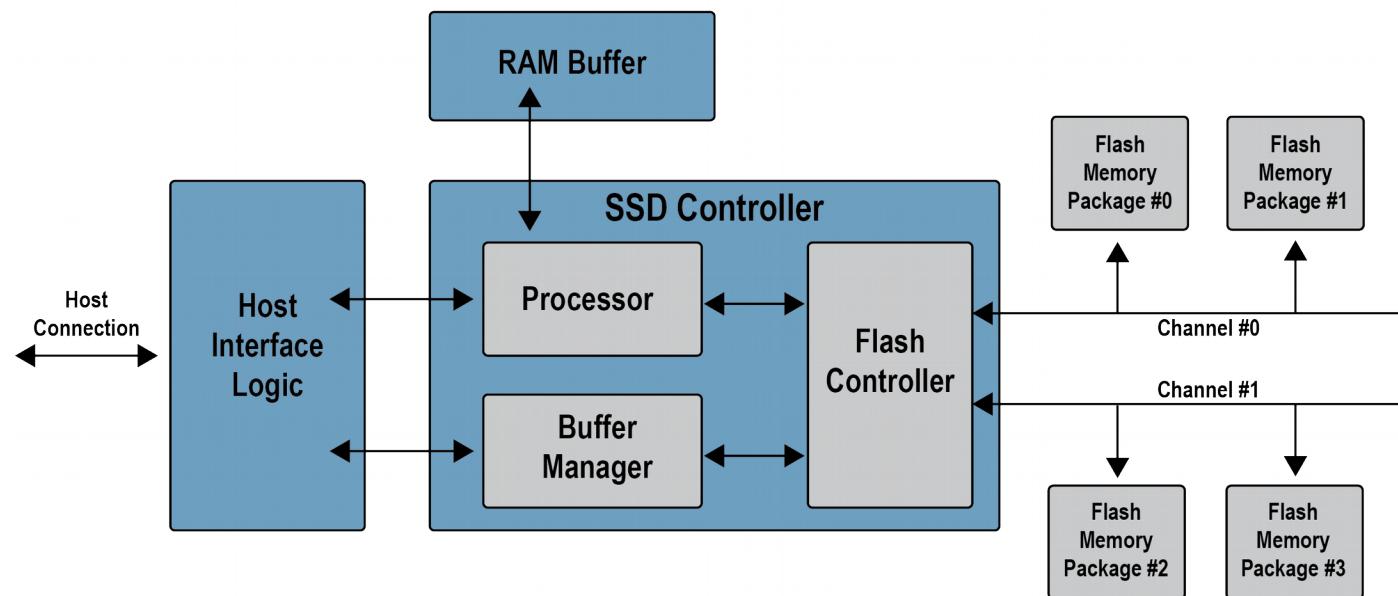
SLC > MLC > TLC  
왼쪽부터 오른쪽 순서로 길다



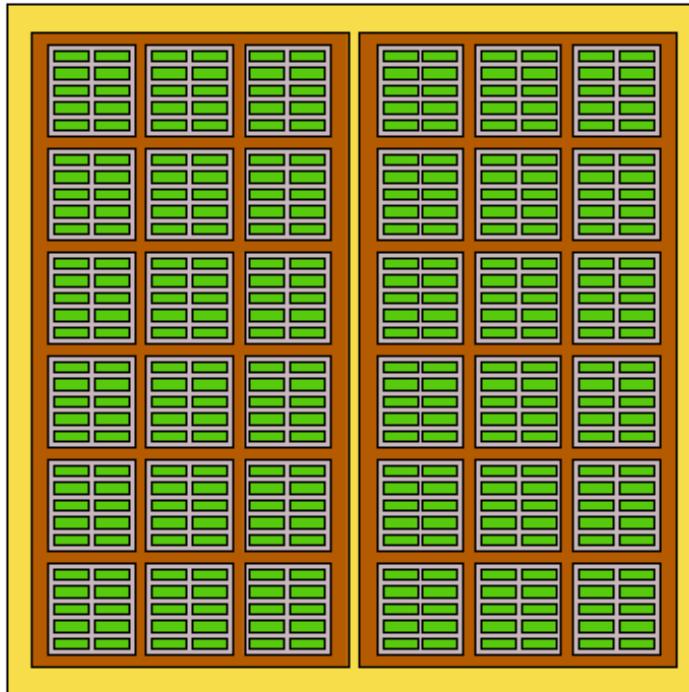
# Architecture of SSD

- Read/Write/Erase
- FTL
- Native Command Queuing

## Solid-State Drive Architecture



# Architecture of SSD



Die  
Plane  
Block  
Page

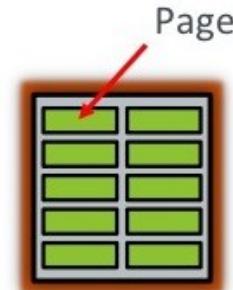
Parameters	Value	Parameters	Value
Number of Channels	16	$T_r$ (in Page)	0.03ms
Chips per Channel	4	$T_p$ (in Page)	0.6ms
Dies per Chip	2	$T_e$ (in Block)	3ms
Planes per Die	2	$T_{transfer}$ (in Byte)	10ns
Blocks per Plane	1,024	Reserved Space	15%
Pages per Block	64	GC Threshold	5%
Page Size	4KB	IO Queue Length	64
Valid Page Buffer	128		

Parameters	Values
Page read to register	20us
Page write from register	200us
Block erase	1.5ms
Read one byte data from register	25ns
Write one byte data to register	25ns
Channel-Chip-Die-Plane-Block-Page	4-4-2-2-2048-64
Page size	2KB

Page Size 는 일반적으로 2KB ~ 4KB 정도  
Block 당 128 ~ 512 개의 Page 를 포함

# Basic Operations

Read : on page size



Write : on page size

Erase : on block size

Operation	Area
Read	Page
Program (Write)	Page
Erase	Block

Overwrite X (impossible) : copy – erase – write

# Basic Operation

## 1. Initial configuration

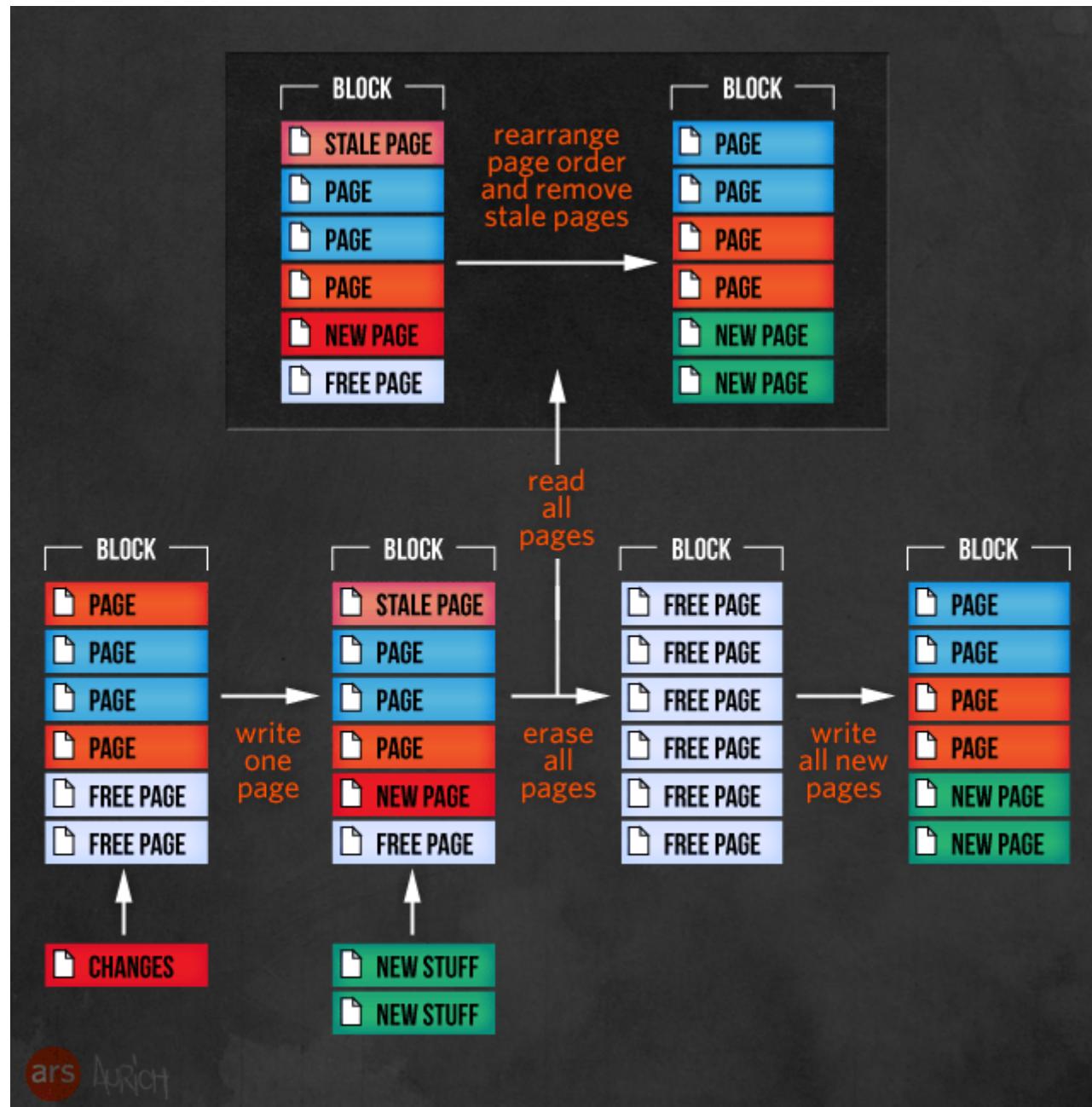
Block 1000 (data)	
PPN	data
0	x
1	y
2	z
3	

Block 2000 (free)	
PPN	data
0	
1	
2	
3	

## 2. Writing a page

Block 1000 (data)	
PPN	data
0	x
1	y
2	z
3	x'

Block 2000 (free)	
PPN	data
0	
1	
2	
3	



# Write Amplification, Wear Leveling

## Write Amplification

Write 는 페이지 사이즈에 맞춰서 (Aligned) 실행

우선 페이지 크기에 맞춰 캐시로 읽어온 후 해당 데이터를 변경한 후 캐시의 데이터를  
다시 플래시 메모리에 기록

페이지 사이즈보다 작은 쓰기는 부가적인 쓰기 (write amplification) 을 동반한다

페이지 사이즈보다 작은 데이터 쓰기는 피하자

작은 데이터의 쓰기는 우선 버퍼링

## Wear Leveling

SSD 는 프로그램 - 삭제 횟수가 제한적

특정한 블록에만 데이터를 읽고 쓸 경우 셀 메모리의 수명 종료

전체 사용 공간이 감소

SSD 전체 메모리 블록의 수명을 골고루 사용하기

이를 위해서 FTL(Flash Translation Layer) 등장

“Write Amplification” 과 “Wear Leveling” 사이의 적절한 타협

# Flash Translation Layer

## Flash Translation Layer, FTL

SSD 와 HDD 는 동일한 호스트 인터페이스를 사용

HDD 는 LBA(Logical Block Address) 를 사용

SSD 컨트롤러는

SSD 내부의 메모리 공간을 효율적으로 사용 + 호스트 인터페이스와 연동을 위해서  
FTL 을 가지고 있음

컨트롤러 역할 2 가지

- 1) Logical Block Mapping
- 2) Garbage Collection

# FTL – Logical Block Mapping

## Logical Block Mapping

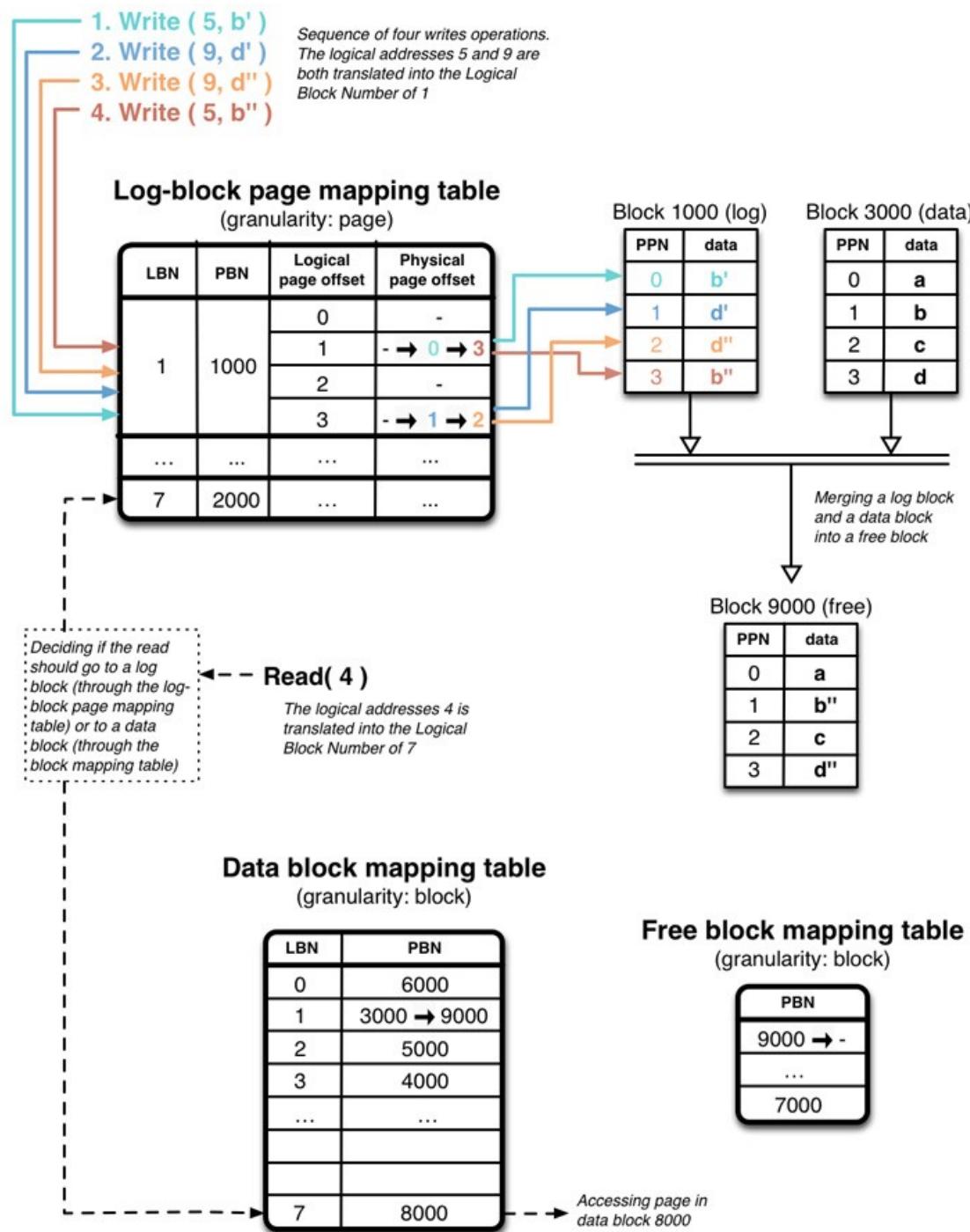
호스트 영역의 논리 주소 (LBA) 를 NAND Flash Memory 의 물리적 주소 (PBA) 로 변환하는데 사용

빠른 처리를 위해 전원이 켜지는 순간 ssd 의 ram 메모리에 로드

두 가지 종류 page level mapping, block level mapping,

그리고 이 둘 사이의 trade-off 를 적절히 이뤄낸 hybrid log-block mapping  
로그 블록이 병합 되기 전에는 로그 블록 , 병합 이후에는 데이터 블록에서 읽기

## Hybrid log-block FTL



# FTL – Garbage Collection

## Garbage Collection

ssd 는 덮어쓰기 불가능

모든 페이지는 셋 중의 한 가지 상태를 지님 (free, stale, data)

데이터 읽기 (25~50 micro second), 쓰기 (250~300 micro second) 는 빠르지만  
삭제 (erase) 하는 과정을 오래 걸림 (1500 ~ 3500 micro second)

background gc(= idle collection), parallel gc(= write 요청이 들어왔을 때 실시 )

background gc 는 foreground task 를 방해할 가능성

( 이를 개선하는 TRIM, Over-provisioning 이 있음 )

parallel gc 는 워크로드가 아주 클 때 사용

Workloads 의미 = 일정한 시간 내에 얼마나 많은 작업이 컨트롤러에 요청되는가  
이를 잘 처리하면 throughput 이 높고 latency 가 낮음

Read Disturb??? hot data, cold data 가 동일한 블록 내에 위치 , 컨트롤러 부하 생성

hot – cold 분리

hot data buffering

불필요한 데이터는 한번에 많이 삭제

# FTL – Garbage Collection

## TRIM

지연된 삭제 (delayed erase)

컨트롤러는 stale 페이지에 data 가 쓰여있다고 인식

Wear leveling 과정에서 이미 삭제된 데이터도 계속해서 복사

이러한 문제점을 해결하는 것이 TRIM

필요없는 논리 공간에 대한 정보를 OS → ssd controller 로 전달

현재 판매되는 대부분의 ssd 디바이스는 TRIM 을 기본적으로 지원

OS, File System, SSD Controller 셋 모두가 TRIM 을 지원할 때만 사용 가능

## Over-provisioning

논리적인 블록보다 물리적인 블록의 수가 더 많도록 해주는 기법

ssd 컨트롤러는 접근 가능하지만 OS 는 접근하지 못하는 ssd 메모리 영역을 확보

쓰기 워크로드를 흡수하는 버퍼공간으로 사용

ssd 성능 향상과 wear leveling 에 도움이 된다 .

# Internal Parallel Processing

## Native Command Queueing

ssd 드라이브의 내부적인 병렬처리 능력을 이용해서 여러 명령을 동시에 처리  
호스트 CPU 가 바쁠 때는 드라이버가 직접 처리하는 명령의 우선순위를 높여주는 것이 예시

## 기타 사항

### \* 보안 향상

secure erase : ( )

advanced error recovery : ( )

end-to-end cyclic redundancy check(CRC check) : ( )

# Internal Parallel Processing

## Internal Parallel Processing

물리적인 한계로 인해 NAND Flash IO 버스는 32~40MB/s 속도로 처리 가능

Parallel 또는 Interleaving 모드로 더욱 빠른 처리

NAND Flash 칩의 다른 위치에서 2 개 이상의 블록을 동시에 읽도록 해주는 것  
“Clustered Block”

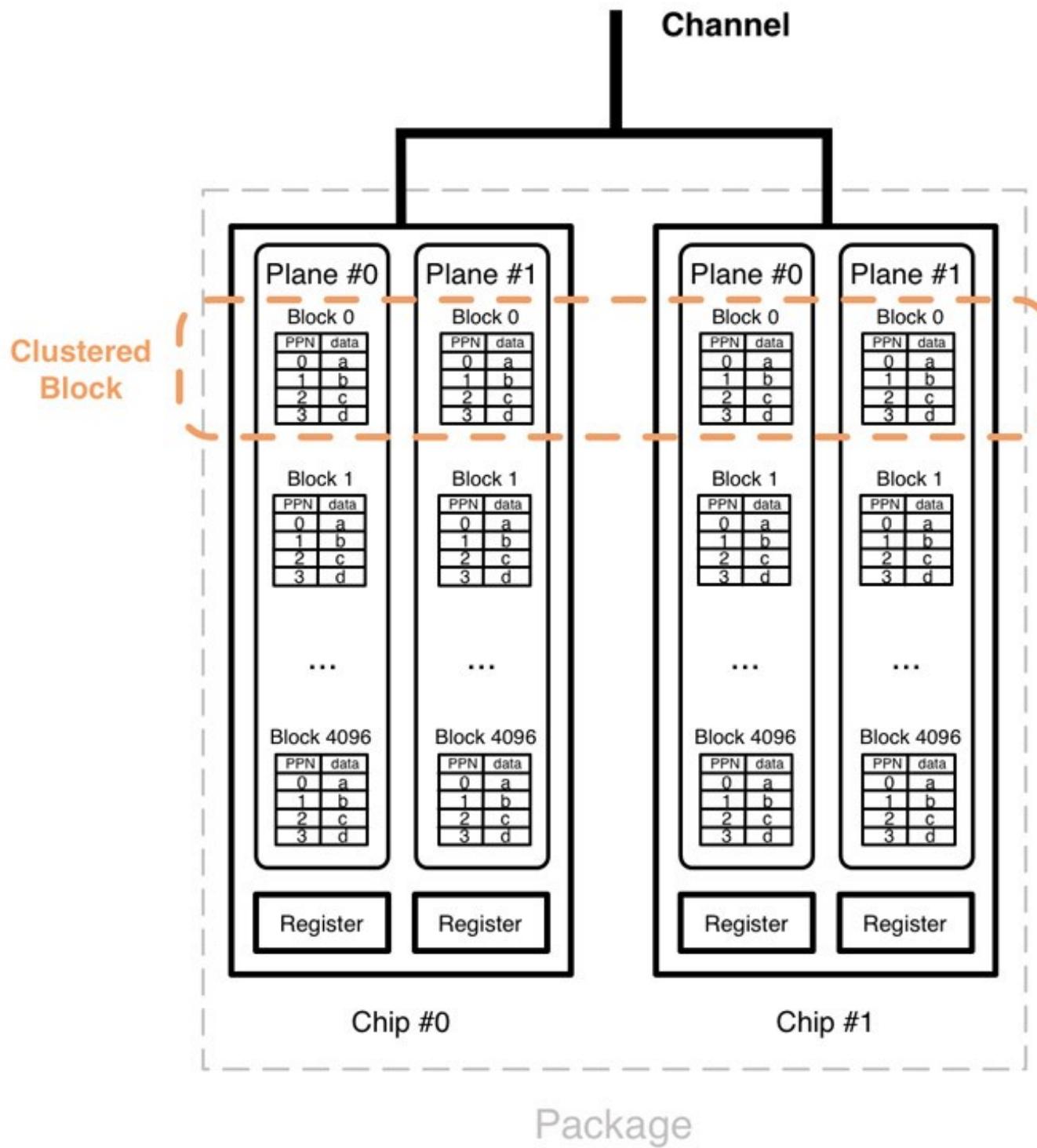
네 가지 수준의 병렬 처리가 가능

Channel-Level

Package-Level

Chip-level

Plane-level



# Internal Parallel Processing

## Optimal WR & RD

Write access pattern 은 sequential & random 두 가지

벤치마킹 결과는 sequential write 가 빠르다고 하지만 (switch-merge 최적화 가능 )

Clustered Block 의 배수 크기의 데이터를 다룰 경우 모든 병렬처리 능력을 사용하면

두 가지 방식 사이의 속도 차이는 거의 없음

빈번하게 작은 크기의 데이터를 쓰기 (write) 할 경우

1) 블록 맵핑을 위한 처리가 많아지고 ram 과 flash 메모리 사이의 동기화 증가

2) copy-erase-write 오퍼레이션이 증가하고

stale 페이지가 전체 물리 공간에 scattered ► internal fragmentation.

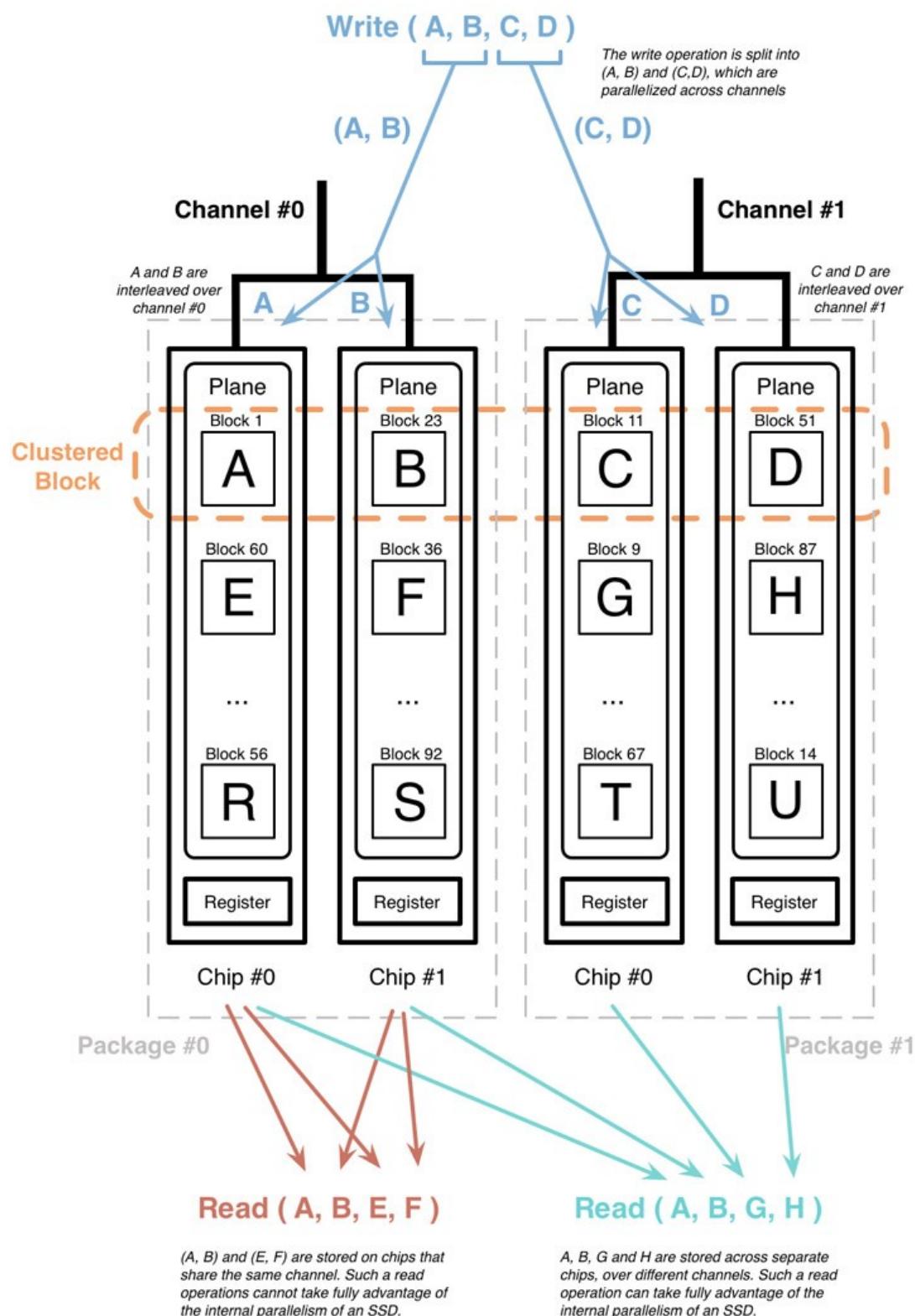
하나의 큰 데이터 쓰기는 많은 동시 작은 데이터 쓰기보다 낫다 .

버퍼링 되지 못하는 작은 쓰기는 멀티 쓰레드로 기록하는 것이 도움이 된다 .

Read 성능 증가를 위해서는 연관된 데이터를 같은 블록에 기록해야 성능 향상

그러나 Read Ahead Buffer 덕분에

단일 쓰레드의 대량 데이터 읽기가 멀티 쓰레드의 소량 데이터 읽기보다 나을 수도



# Optimizing System for SSD

## 시스템 최적화

Partition Alignment : 포맷하거나 운영체제를 새롭게 설치할 때  
SSD 메모리 블록 크기에 맞춰서 설치하기

File System Parameter : Activate TRIM

vm.swappiness = 0

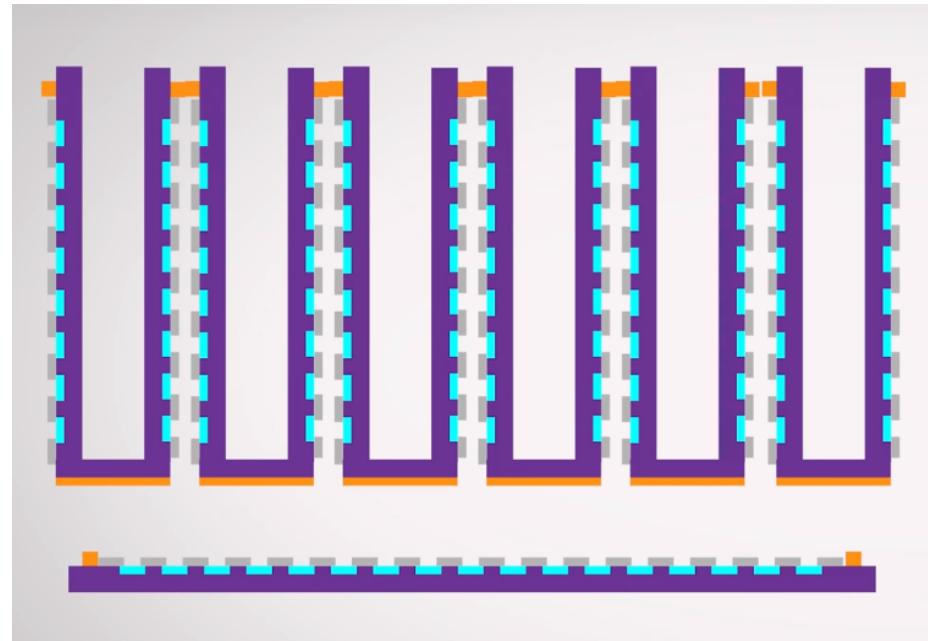
temporary files 는 최대한 RAM 에만 기록하기

# 3D XPoint

## Previous Storage Technology

### 3D NAND

NAND Flash Memory 를 수직으로 접은 것과 같은 모양으로  
메모리의 집적도를 크게 향상 시킬 수 있는 기술



# 3D XPoint

= 3D CrossPoint

## 3D XPoint™ Technology: An Innovative, High-Density Design

### Cross Point Structure

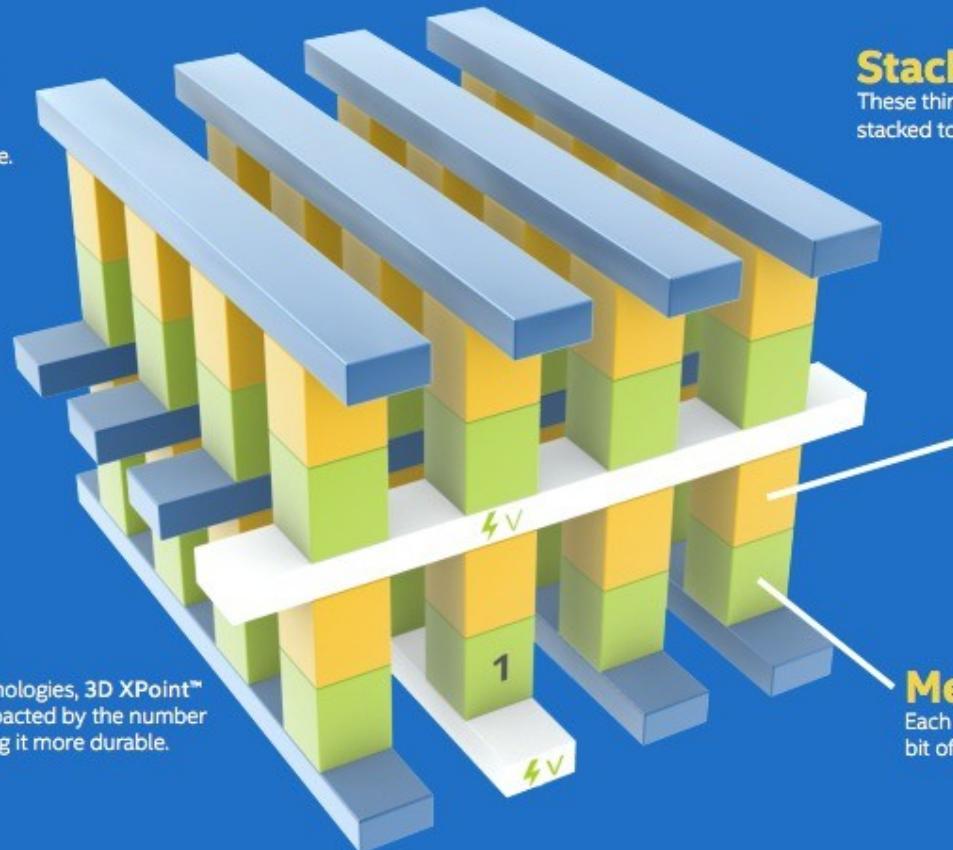
Perpendicular wires connect submicroscopic columns. An individual memory cell can be addressed by selecting its top and bottom wire.

### Non-Volatile

3D XPoint™ Technology is non-volatile—which means your data doesn't go away when your power goes away—making it a great choice for storage.

### High Endurance

Unlike other storage memory technologies, 3D XPoint™ Technology is not significantly impacted by the number of write cycles it can endure, making it more durable.



### Stackable

These thin layers of memory can be stacked to further boost density.

### Selector

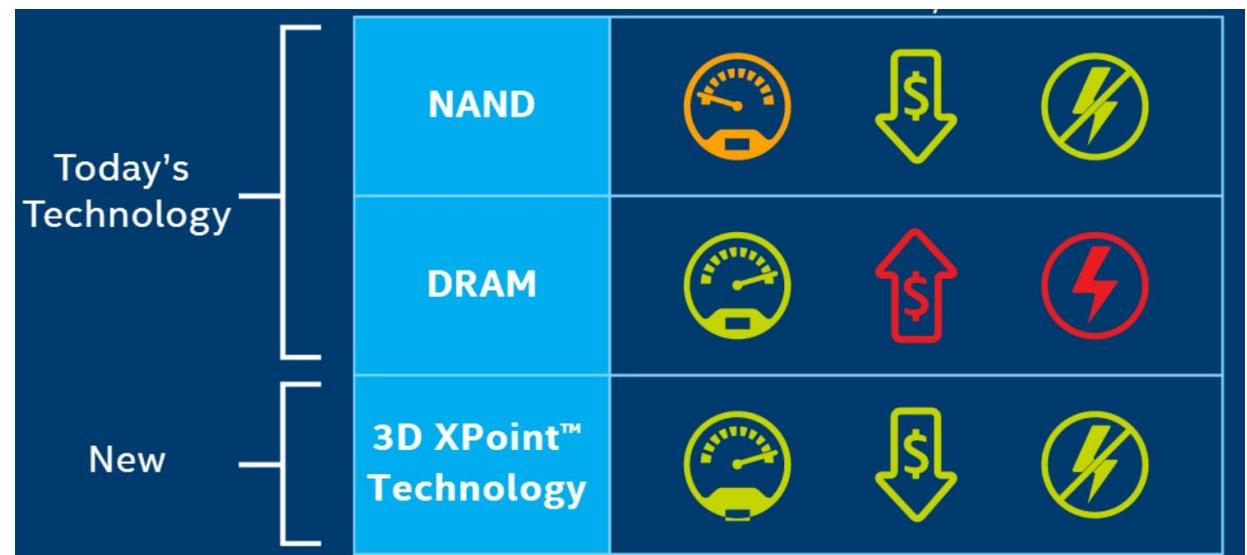
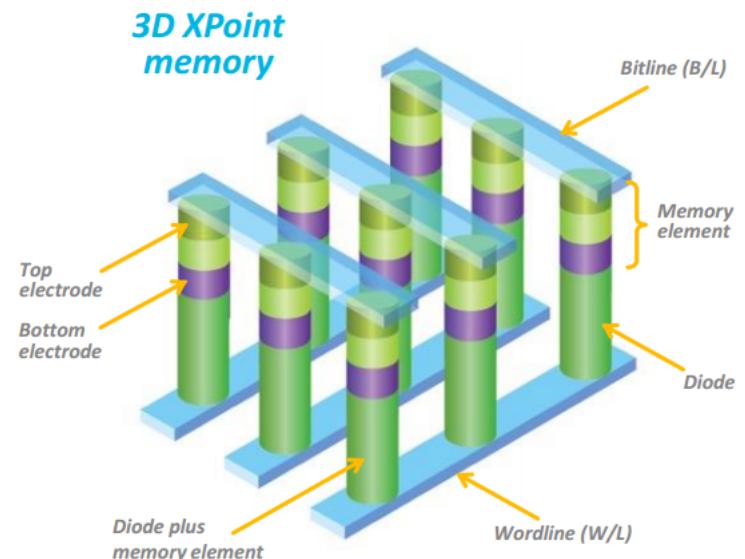
Whereas DRAM requires a transistor at each memory cell—making it big and expensive—the amount of voltage sent to each 3D XPoint™ Technology selector enables its memory cell to be written to or read without requiring a transistor.

### Memory Cell

Each memory cell can store a single bit of data.

# 3D XPoint

- Compared to NAND Flash
  - Faster
  - Low Latency
  - Cost-effective
  - Stackable
  - More Memory Density (= Better Capacity)
  - \* Read/Write on page size (without block size caching)
- Non-Volatile Memory
- Portable with existing hardware interfaces



# NAND Flash SSD vs 3D XPoint Optane (as a Secondary Memory)

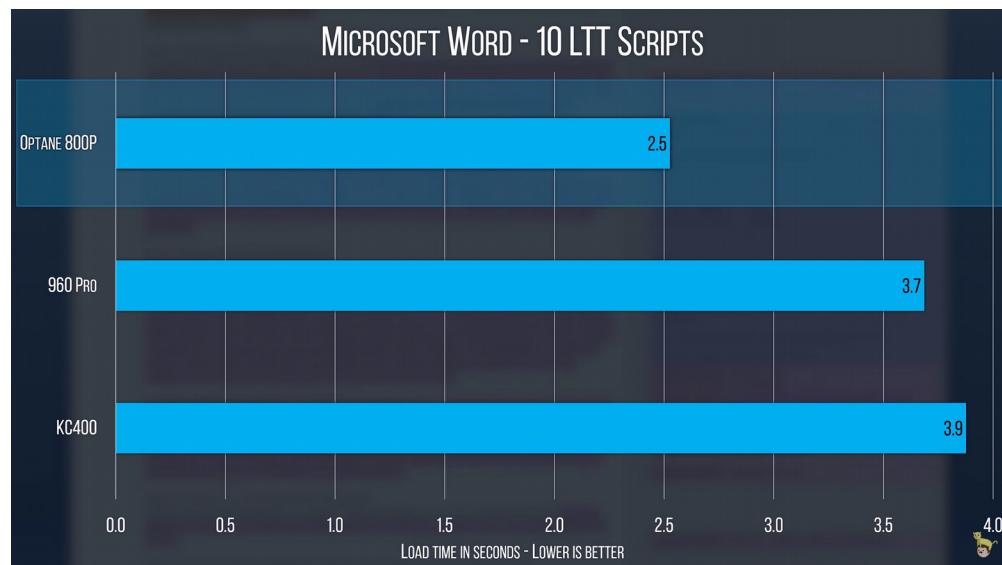
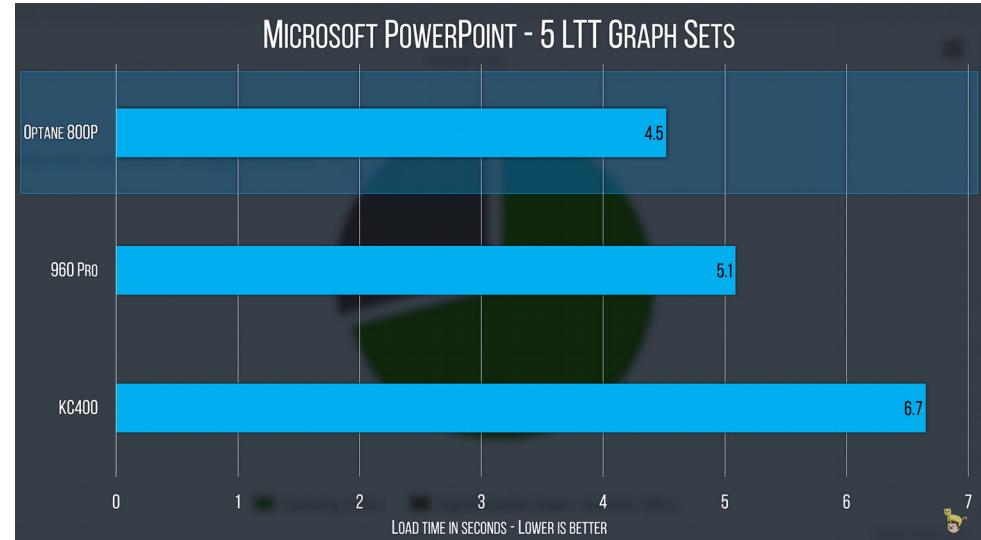
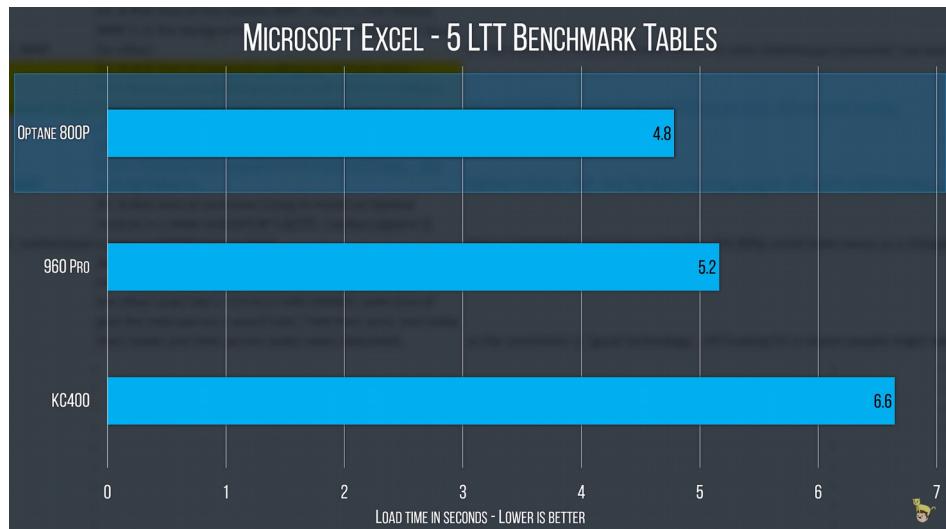
Samsung 960 Evo Pro  
(NAND Flash SSD)

vs

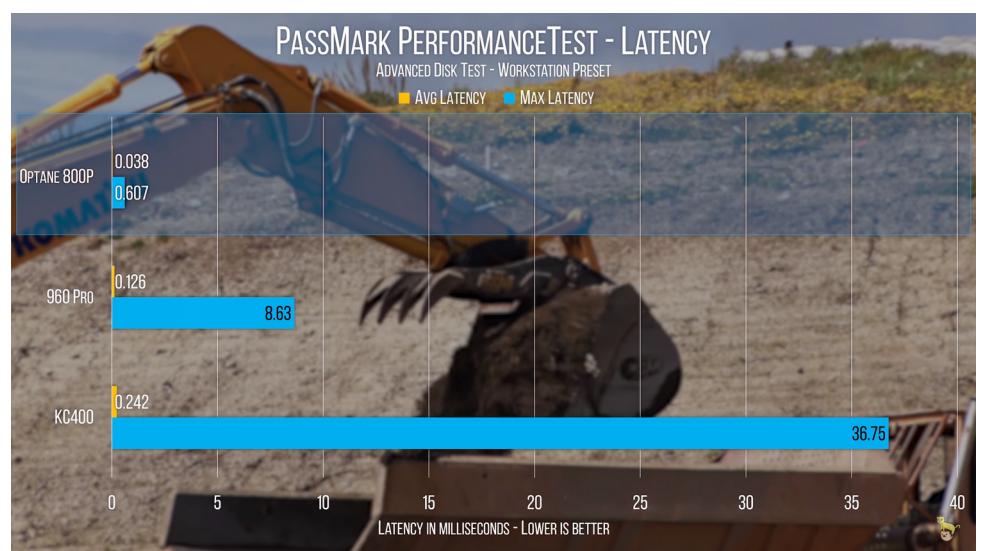
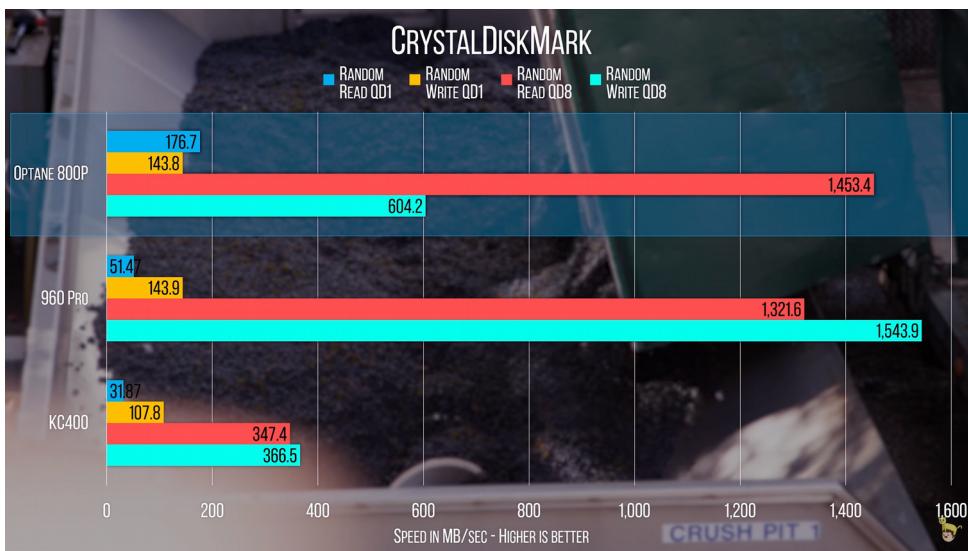
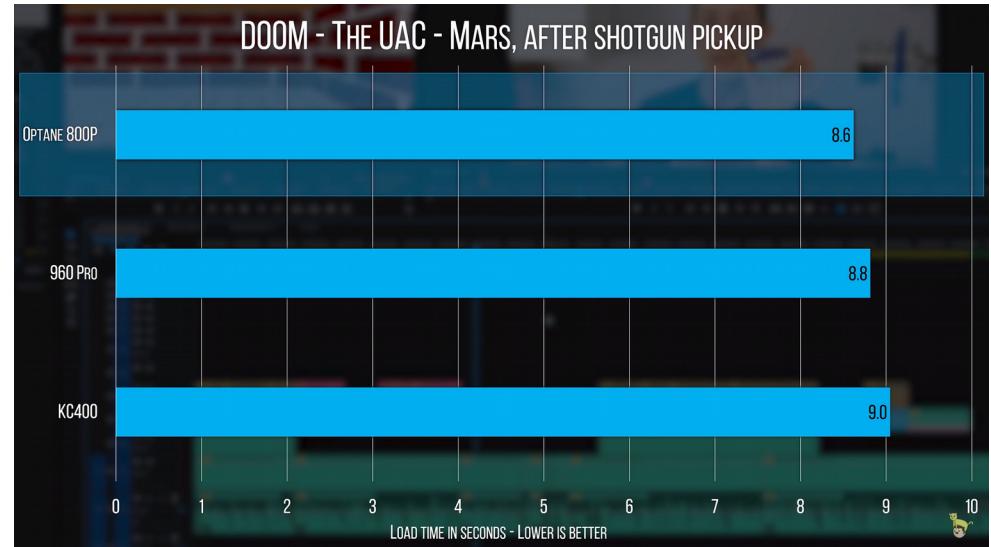
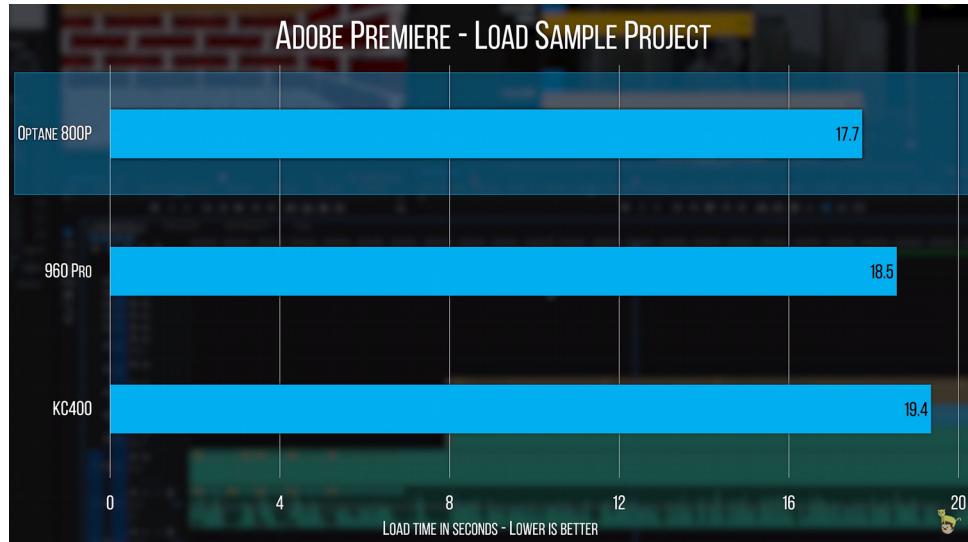
Intel Optane 800 P  
(3D Xpoint Optane)

	SAMSUNG 960 PRO	INTEL OPTANE 800P
PCI-E LANES	x4	x2
BANDWIDTH	3,940 MB/SEC	1,970 MB/SEC

# NAND Flash SSD vs 3D XPoint Optane (as a Secondary Memory)

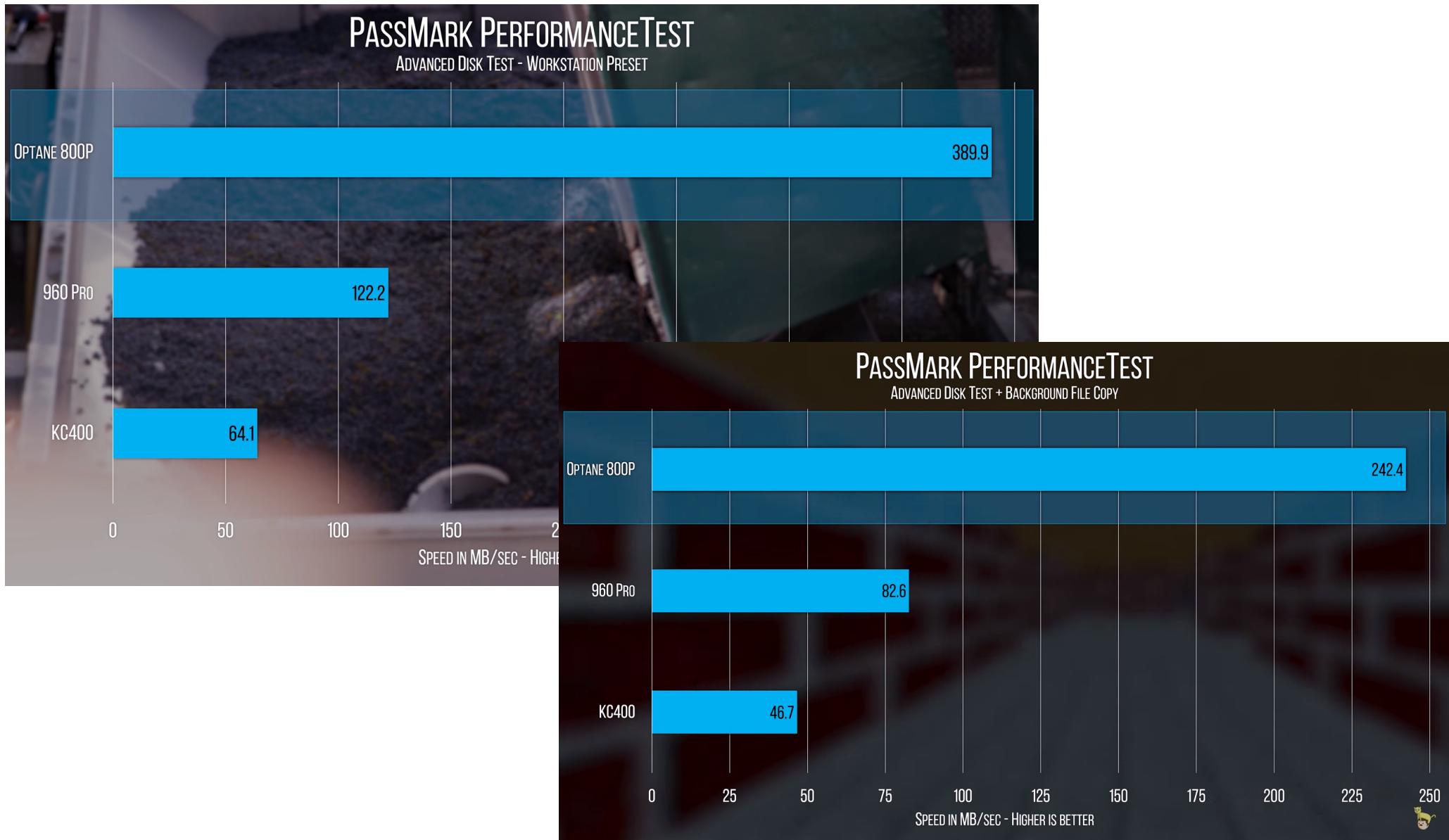


# NAND Flash SSD vs 3D XPoint Optane (as a Secondary Memory)



Queue Depth : the number of pending input/output(I/O) requests for a volume

# NAND Flash SSD vs 3D XPoint Optane (as a Secondary Memory)

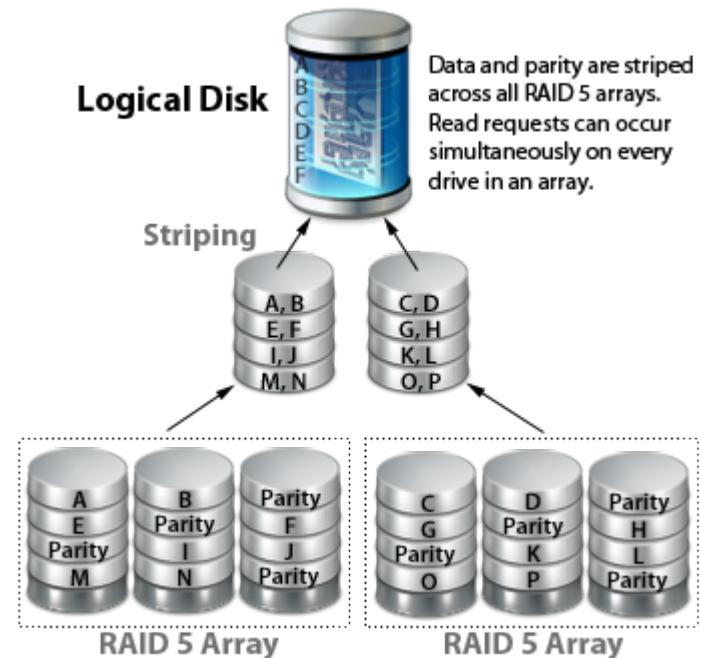
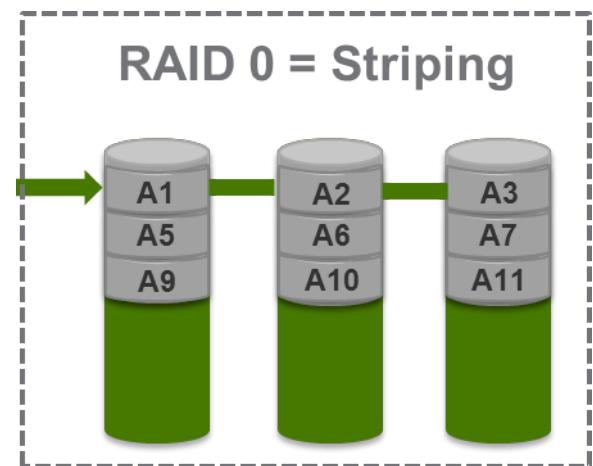


# NAND Flash SSD vs 3D XPoint Optane (as a Secondary Memory)

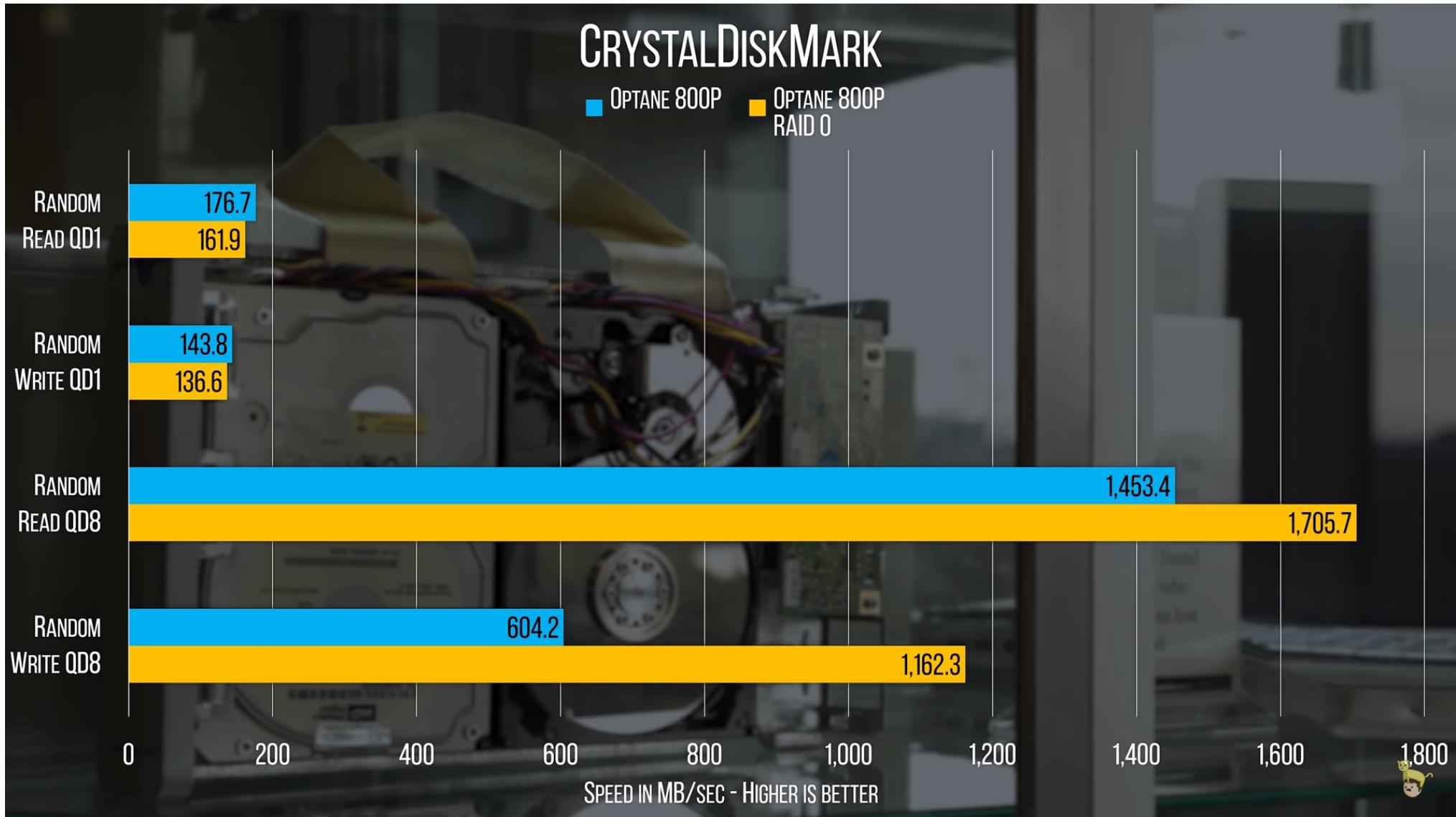
## RAID

어떤 데이터를 논리적으로는 하나의 통합된 파일로 다루지만 물리적으로는 여러 개의 디스크에 분산시켜서 저장하므로써 데이터에 대한 latency 를 낮출 수 있다

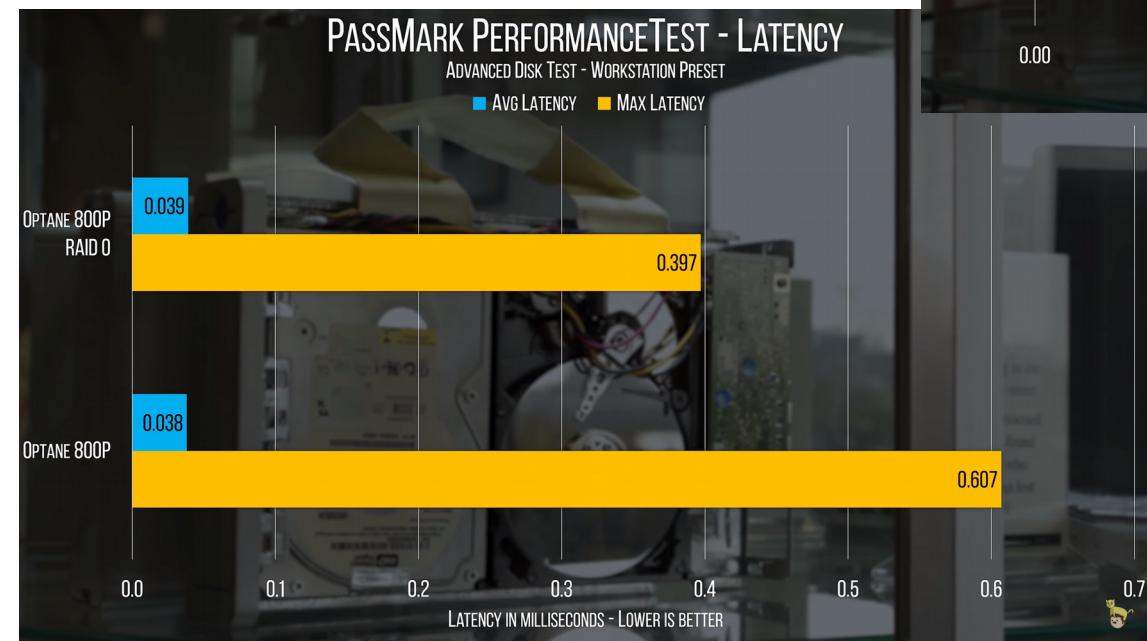
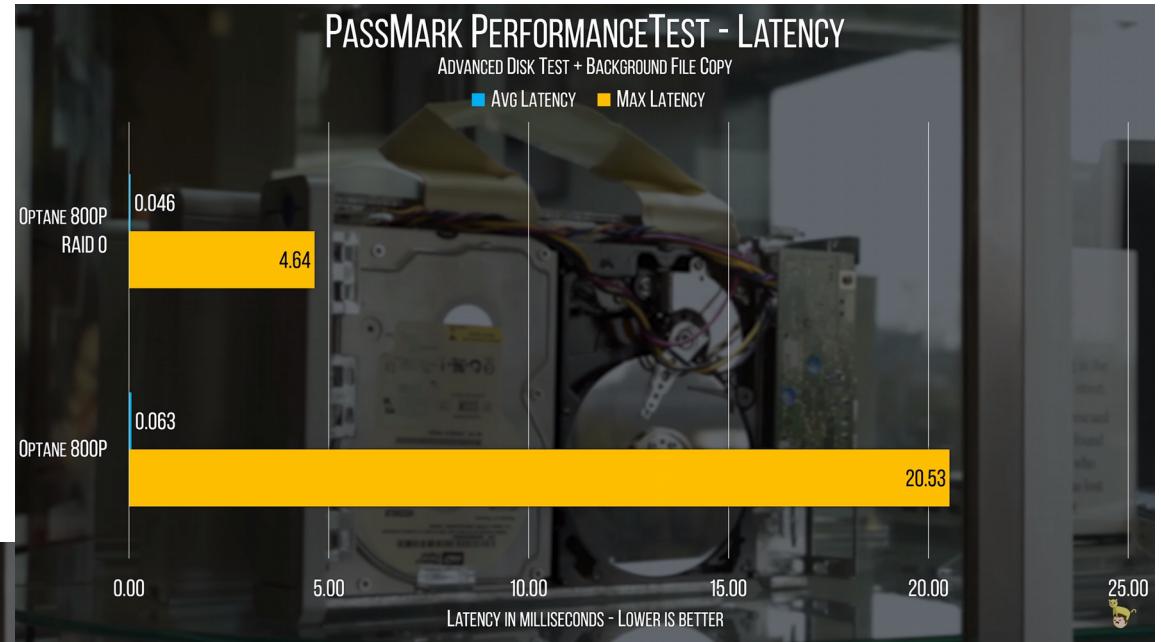
Parity 정보도 RAID 로 분산 저장하므로써 데이터의 안정성을 높일 수 있다



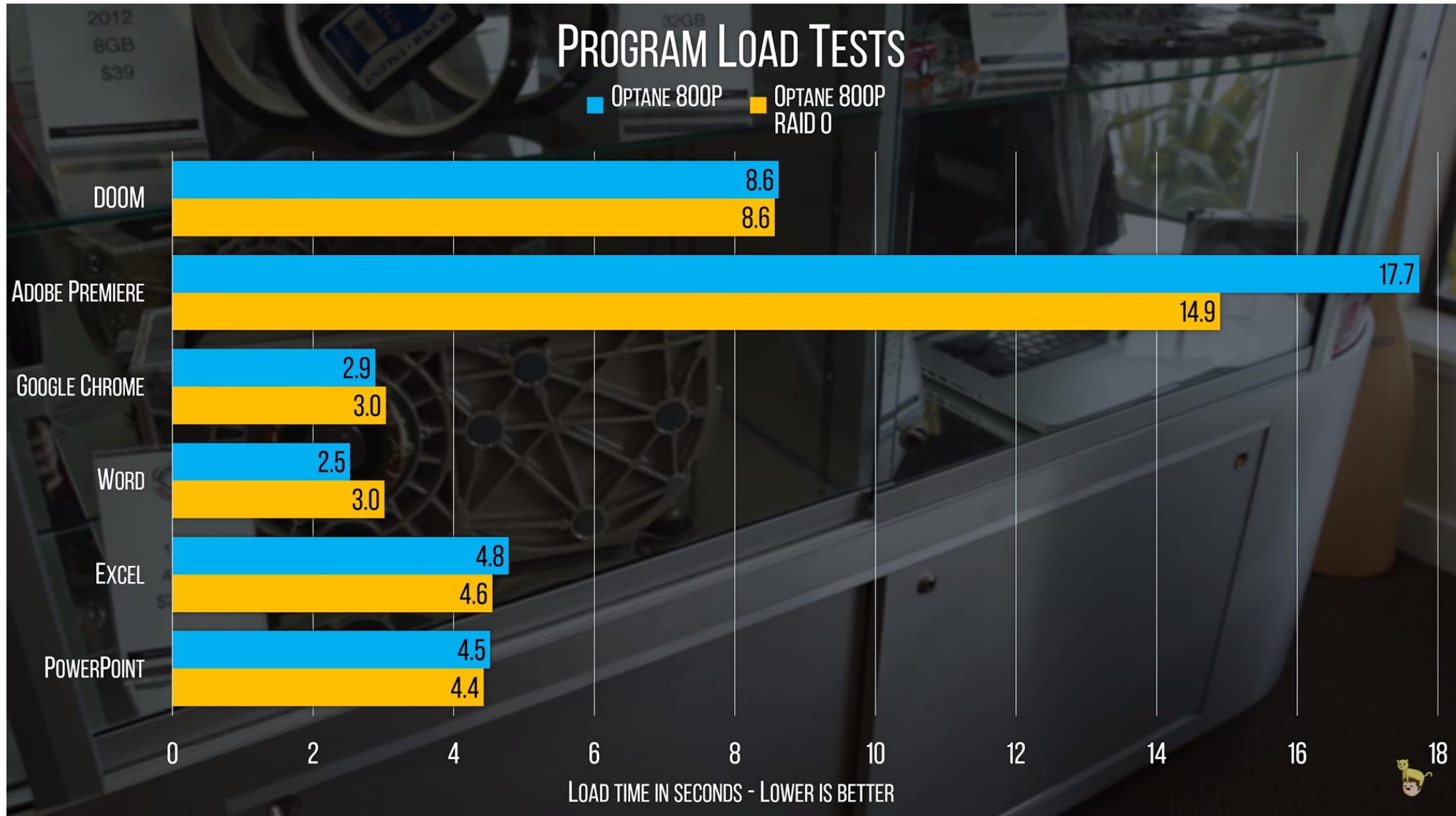
# NAND Flash SSD vs 3D XPoint Optane (as a Secondary Memory)



# NAND Flash SSD vs 3D XPoint Optane (as a Secondary Memory)



# NAND Flash SSD vs 3D XPoint Optane (as a Secondary Memory)



# NAND Flash SSD vs 3D XPoint Optane (as a Secondary Memory)

이를 통해 알 수 있는 사실

3D Xpoint 기술을 적용한 Intel Optane Memory 를 보조 메모리로 사용할 경우에  
일반 사용자 (End Users) 는 별로 흥미를 느끼지 못하겠지만

Read-Intensive 한 서버를 운영하는 비즈니스 사 입장에서는  
본 기술을 자사에 도입하는 것을 고려해볼 만하다

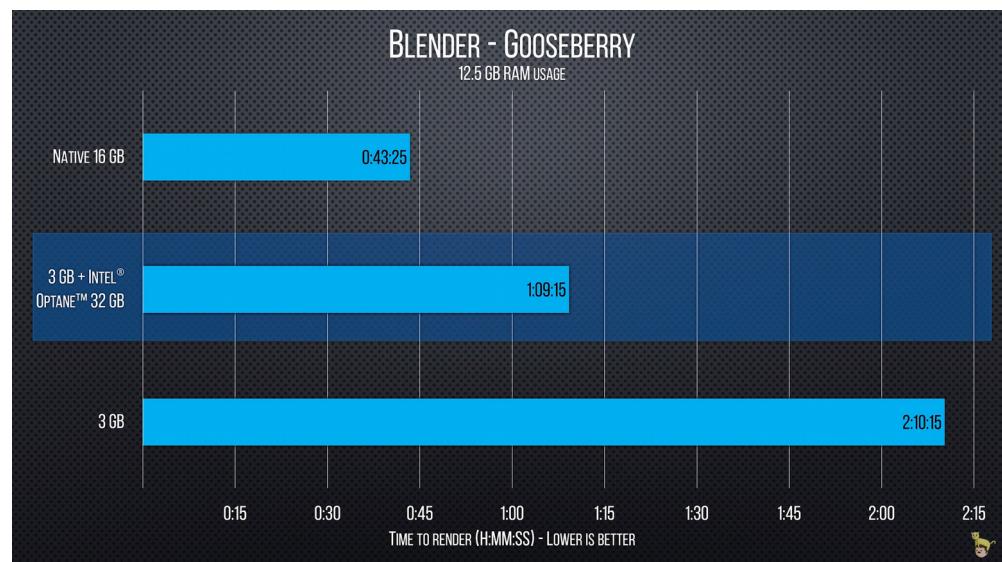
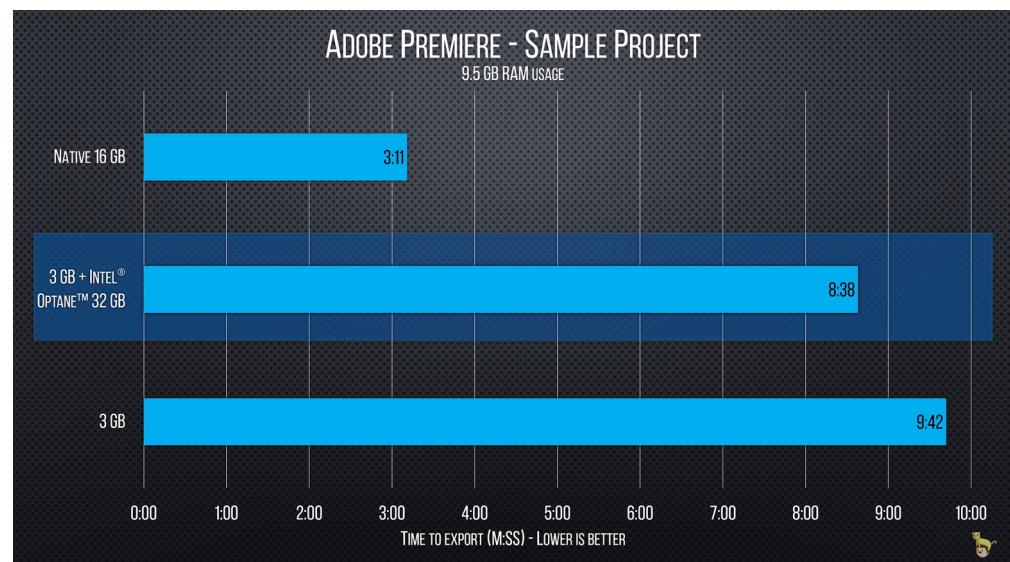
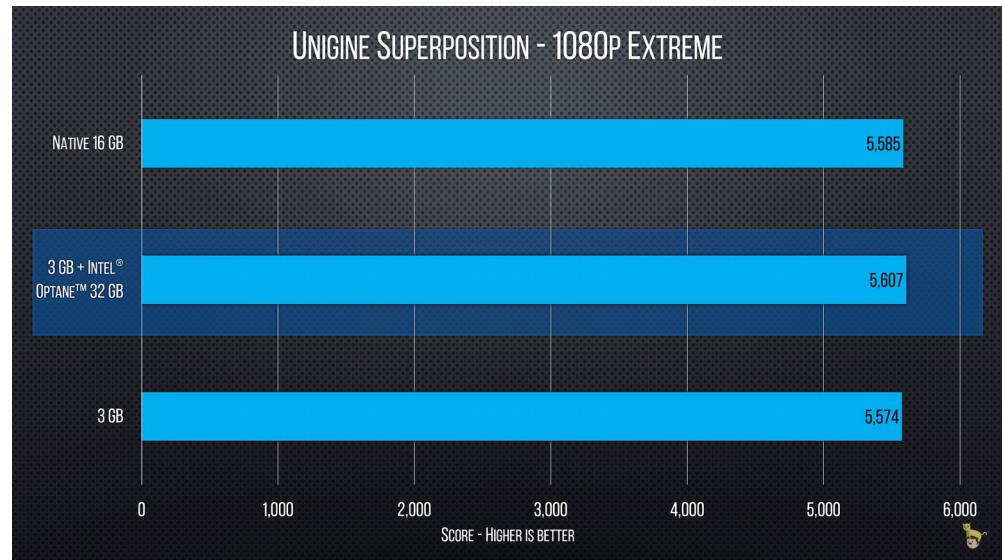
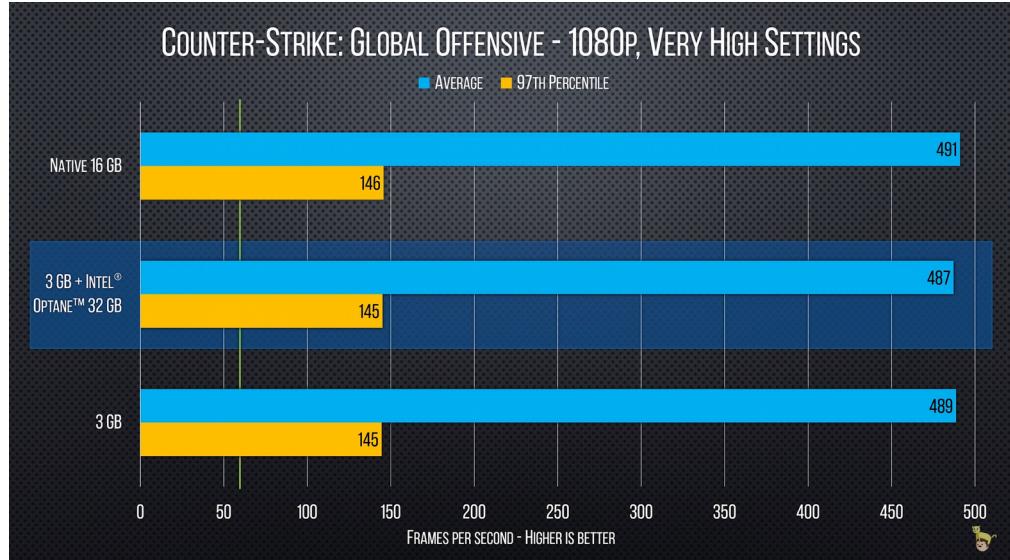
# NAND Flash SSD vs 3D XPoint Optane (as a Main Memory)

(Condition 1) 16GB DDR4 RAM is the native main memory.

(Condition 3) 3GB parameter is set by windows 10 option to limit RAM usage.

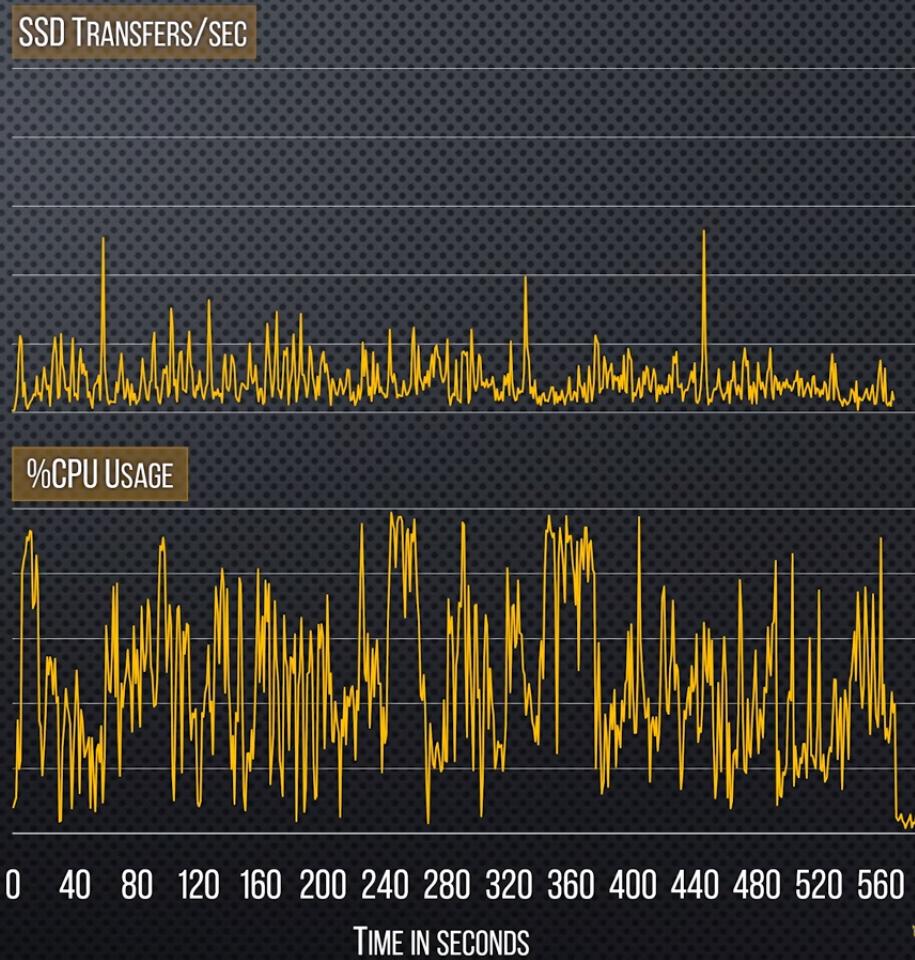
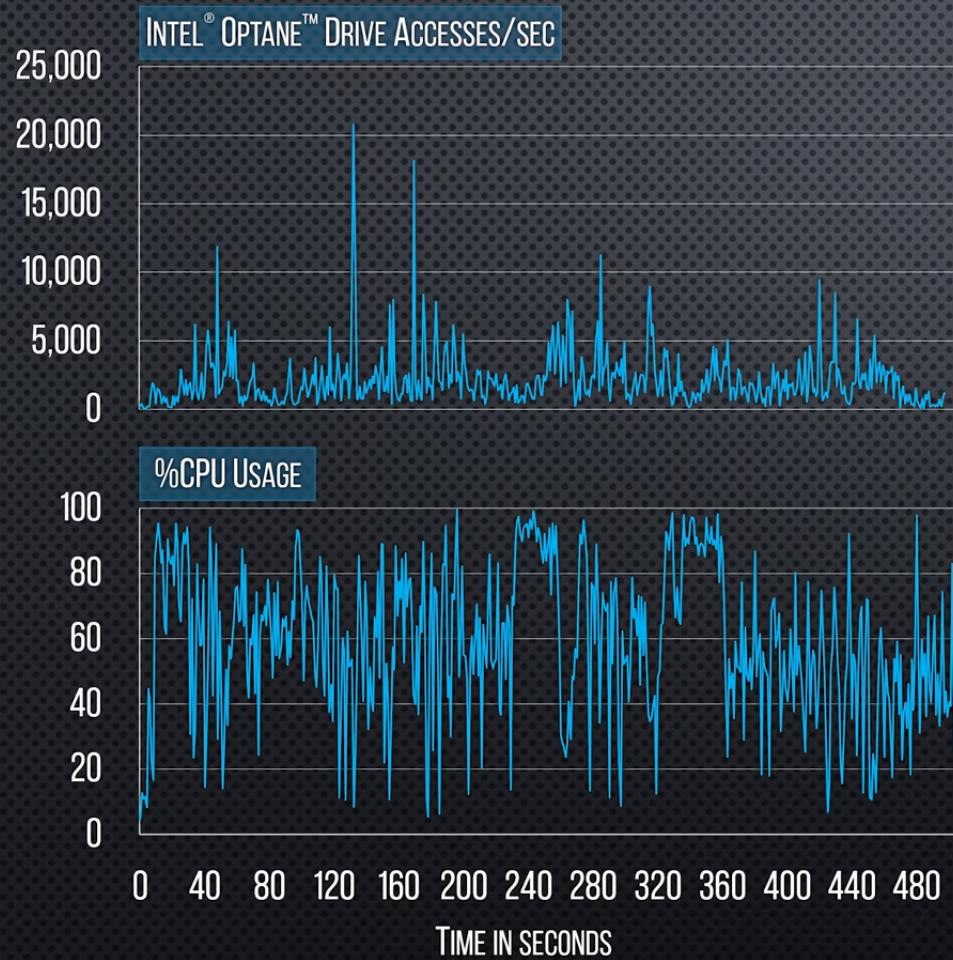
(Condition 2) 3GB + Optane 32GB is using dual channel  
limited native RAM usage and Intel Optane 32GB Memory.

# NAND Flash SSD vs 3D XPoint Optane (as a Main Memory)



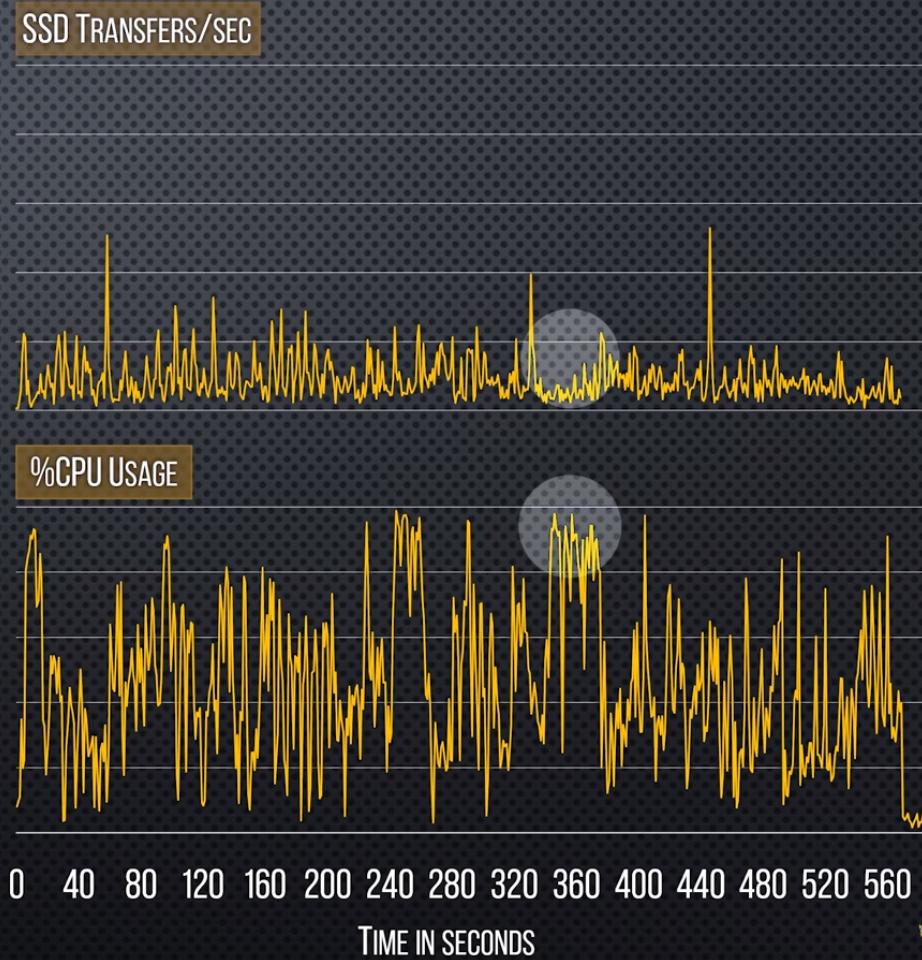
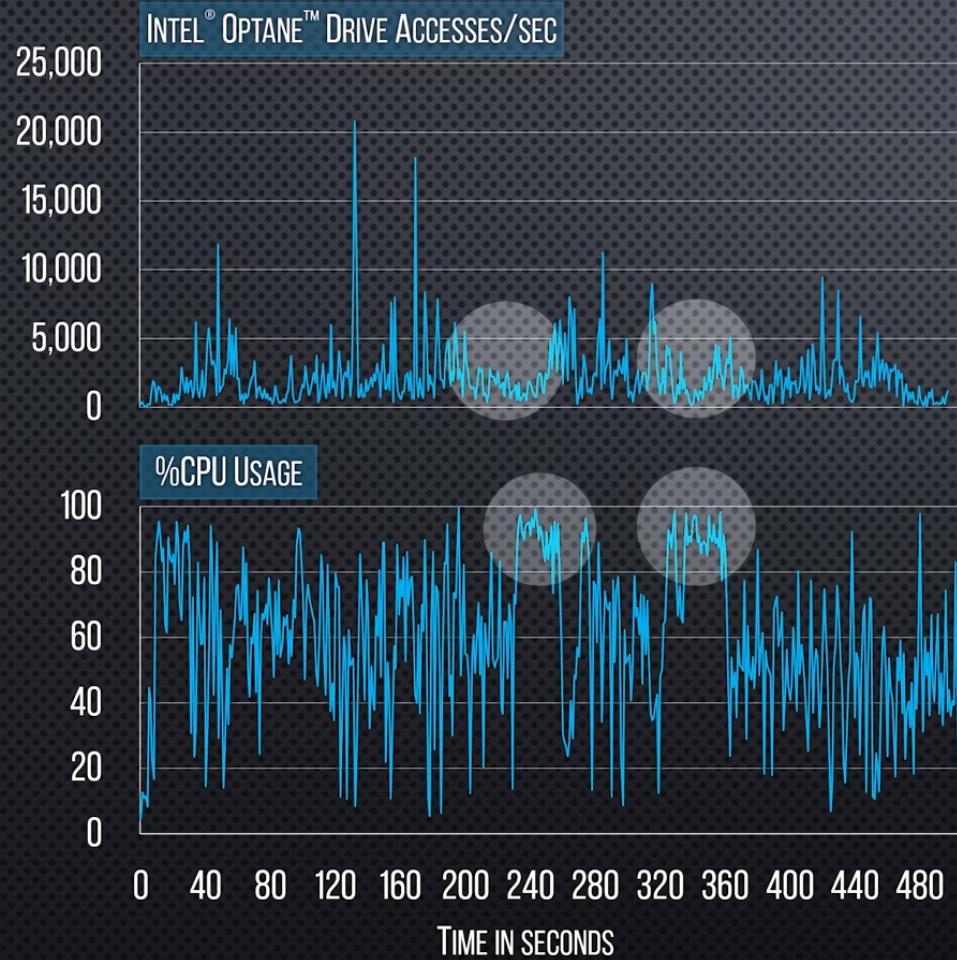
# NAND Flash SSD vs 3D XPoint Optane

## ADOBÉ PREMIERE – SAMPLE PROJECT EXPORT (4K)



# NAND Flash SSD vs 3D XPoint Optane

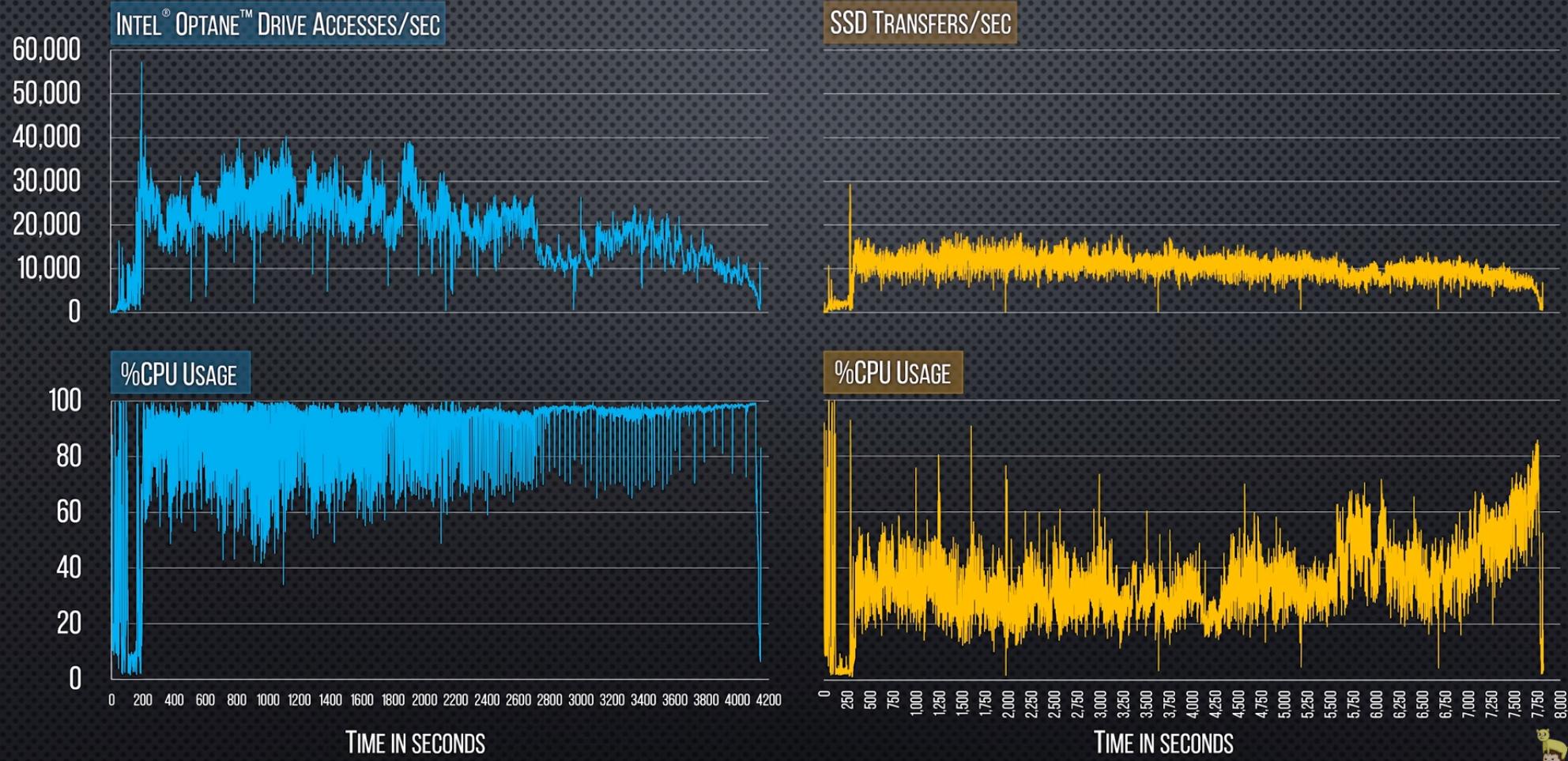
## ADOBE PREMIERE – SAMPLE PROJECT EXPORT (4K)



위의 결과를 자세히 보면 CPU 100% 일 때 Drive Access 수치가 0에 가까워짐을 알 수 있다

# NAND Flash SSD vs 3D XPoint Optane

## BLENDER – GOOSEBERRY RENDER



보다 적은 Driver Access Bottleneck으로 CPU에 보다 많은 태스크를 요청할 수 있었다

# NAND Flash SSD vs 3D XPoint Optane (as a Main Memory)

이를 통해서 알 수 있는 사실

3D Xpoint 기술은 지금 당장 도입하기에도 손색이 없지만  
기술적으로 보다 발전시켜서  
데이터가 현재보다 훨씬 증가하고 워크로드가 과중해지는  
미래에 맞춰서 적용하면 좋을 기술



# SSD Review

1. SSD 의 수명 제한 : NAND Flash Memory Cell 을 기반으로 하므로 P/E Cycle
2. Page 와 Block : 기본적으로 Overwrite 은 불가능하고
  - Read/Write 는 페이지 단위로 , Erase 는 블록 단위로
  - 그리고 Read/Write 시에도 해당 블록을 먼저 RAM 메모리로 옮겨놓아야 함
  - Aligned Read / Aligned Write 라고 한다
3. Wear Leveling : SSD 의 모든 메모리 블록을 고르게 사용하기
4. FTL(Flash Translation Layer) : 플래시 메모리를 오래된 인터페이스에서 사용하기
  - 4 – 1. Log Block Mapping, Data Block Mapping
    - 이를 통해서 random I/O 를 sequential I/O 처럼 실행하는 이점
  - 4 – 2. Garbage Collection 을 통해서 stale 페이지를
    - 효율적으로 free 페이지로 만들어서
    - 명령 처리 시에 latency 적게 발생하도록 보완
5. Access Pattern
  - 작은 데이터 WR 는 버퍼링 후 한꺼번에 처리
  - RD 성능 향상을 위해서 관련된 데이터는 동일한 블록에 저장
  - RD/WR 요청은 분리해서 실행 -> Mass Sequential 처리를 권장

# SSD Review

6. Invalidate Obsolete Data in batch : Erase 오퍼레이션은 버퍼링 후 대용량 처리

7. Optimal WR

Random writes are not always slower than sequential writes

기록하는 데이터 크기가 Clustered Block 크기의 N 배수 (N 은 정수 ) 라면

Internal parallel capacity 를 최대한 활용 가능

8. Optimal RD

Reading Mass data with single thread shows better throughput than  
Multiple thread processing.

9. 소량의 데이터 WR 를 버퍼링 / 그룹핑 불가능할 때만 multi-thread 로 실행

10. Hot / Cold Data Separation : 효율적인 garbage collection 을 위해

Hot Data Buffering 을 활용

11. 시스템 최적화

PCIe 또는 SAS 인터페이스 사용

Over-provisioning = 여분의 물리 공간을 남겨서 wear leveling 에 사용

Activate TRIM Command

Partition Alignment

# More...

Coding for SSD : <http://www.goossaert.com/>

개발자를 위한 SSD : <http://tech.kakao.com/2016/07/13/coding-for-ssd-part-1/>

Solid-state revolution: in-depth on how SSDs really work :

<https://arstechnica.com/information-technology/2012/06/inside-the-ssd-revolution-how-solid-state-disks-really-work/>

Storage for DBAs : <https://flashdba.com/storage-for-dbas/>

3D XPoint - Reality, Opportunity, and Competition(Video) :

<https://www.youtube.com/watch?v=hq1Iz5JKbAA>

Why Did Intel Even Make This? – Optane 800P SSD :

<https://www.youtube.com/watch?v=oWqO36Zj65k>

Save Money on RAM.. with Optane?? :

<https://www.youtube.com/watch?v=cwy4ujt0qHM&t=408s>