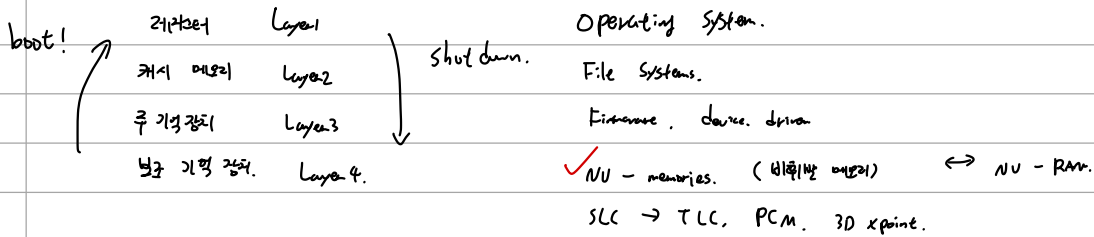


시뮬레이션.

제품을 완벽히 만들어 내고, 특징 기능 요구사항을 구명해서 성능을 끌어올릴 것.

스도리 시스템.

컴퓨터 구조부터 시작된다.



HDD → SSD로 변경시 문제점 (low-level)

기본, SATA 규격에 맞게 문명에서, 하드웨어로 들어 오기때, SSD는 다른 규격의 장치이다.

가변형 문명에서, 하드웨어에서 작동해야 한다.

Firmware, Controller를 만들어야 한다. (ex) SSD Controller driver, SATA 인터페이스, F2FS와 다른 NTFS, EXT4 여타의 시스템. Application layer이 중요성

그리고 Low-level I/O, Controller를 이런 상 가지는 프레임 워크, 소프트웨어를 개발한다.

1. 결국, software의 I/O가 느려져. // SSD를 기반으로 만든 소프트웨어 성능을 떨어뜨리게. (Application layer)

2. 컴퓨터 아키텍처의 구조적인 문제 때문. // 가장 낮은 layer인 보조 기억 장치를 바꾸고, 소프트웨어를 개발한다. 비휘발성! 보조 기억 장치의 I/O가 줄어들어서 해결이 안된다. ex) 컨트롤러

TLC의 지명성이 더 높지만, 왜 사용하냐 // Controller 때문.

program cycle limit: 쓰기 횟수

SLC처럼 한번에 3bit read, 리드 시간이 ↓. (TLC의 성능 문제)

Fusion memory: SLC cache + TLC. 데이터 접근시 SLC에 진입.

NV-RAM. // 비휘발성 메모리. 즉, 보조 기억 장치

데이터 크기 변조 가능. 속도 ↑ // 결국 NV-RAM의 용량을 용해, 그리고, FTL, 등 어떻게 활용?

DRAM < PRAM < NVM.

⊗ 파일 크기 ↑ 디스크 할당 크기의 미비!

레거시 메모리 구조

↳ 결국 파일 I/O 효율성에 맞게 데이터가 접근이 되지 않거나,

그리고 결국, 기존의 파일 시스템이 비효율적이게!