CS 423
Operating System Design: Persistence: Storage Devices & RAID
Apr 7

Ram Kesavan

LOGISTICS

C4 paper: RAID

MP3 due by Apr 15, 11:59 CT

Deadline for MP2 resubmission: Apr 7 11:59 CT

Any one of MP2 or MP3 can be resubmitted

Midterm regrade deadline: Apr 7 11:59 CT

AGENDA / LEARNING OUTCOMES

Basics of persistence

Storage device abstraction
- Read/Write from/to the media
- Media characteristics

Connectivity to storage device(s)

RAID

THE FILE SYSTEM

The file system is an abstraction to:

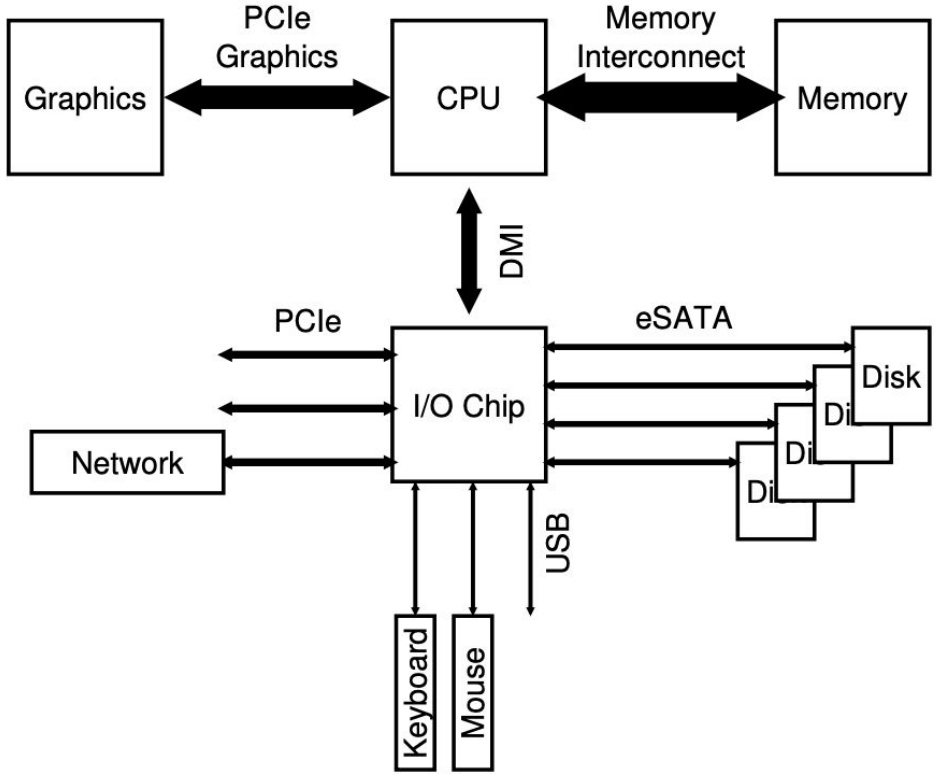
- Hide physical complexity of storage devices
- Present an API to
  - create & delete a file system
  - mount & unmount a file system
  - organize user data into the file system
    - file, directory/folder, etc.
  - read from and write to the file system

File system uses metadata to make this happen

single system vs distributed

Bottom-up & top-down approach to this topic

ARCHITECTURE DIAGRAM



DMI: direct media i/f

- dedicated graphics
- PCIe
  - network, monitor
  - NVMe SSD
- USB
  - kb, mouse, etc
- eSATA/SCSI
  - HDD, SSD

STORAGE DEVICE: ABSTRACTION

Main Components

- storage space: flash or magnetic media
- memory: buffer to stage data to/from device
- controller: CPU + registers (status/cmd/etc)

Data is transferred in units of blocks

- Logical Block Address (LBA)

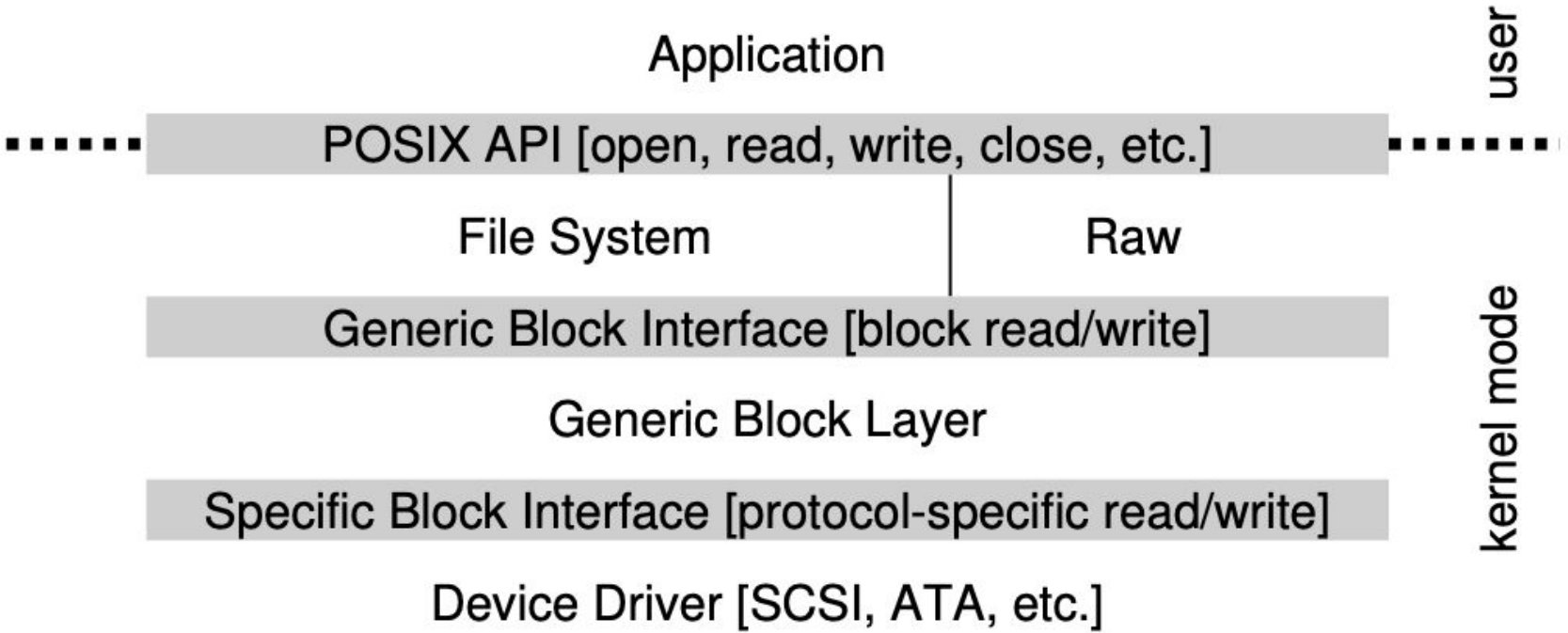
OS communication with device

- Programmed I/O vs DMA
- Completion: interrupt vs polling

Device Drivers

STORAGE DEVICE: WRITE/READ

FILE SYSTEM STACK

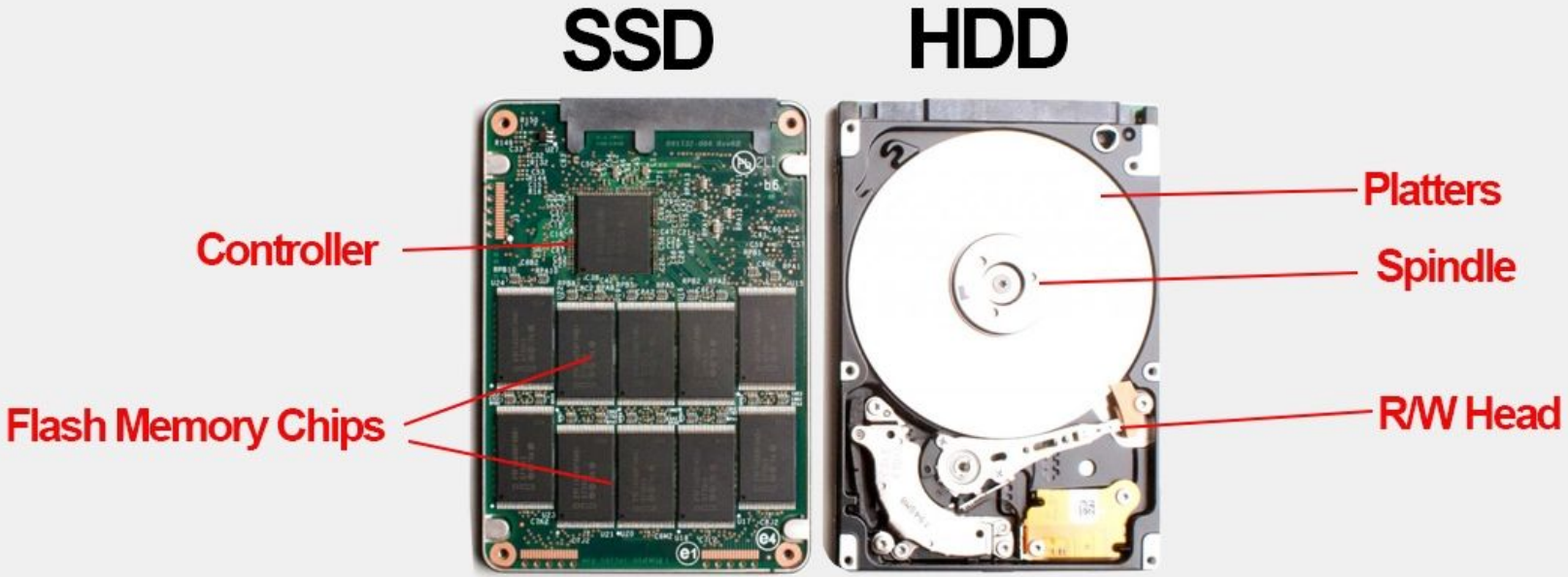


STORAGE DEVICE MEDIA

Magnetic tape
Magnetic floppy disks (obsolete)
Magnetic spinning disks (hard disk drive aka HDD)
    Various capacities & speeds
NAND Flash memory (solid data drive aka SSD)
    Various capacities & speeds
Optical CDs, DVDs

Future research: ceramic, quartz glass, DNA

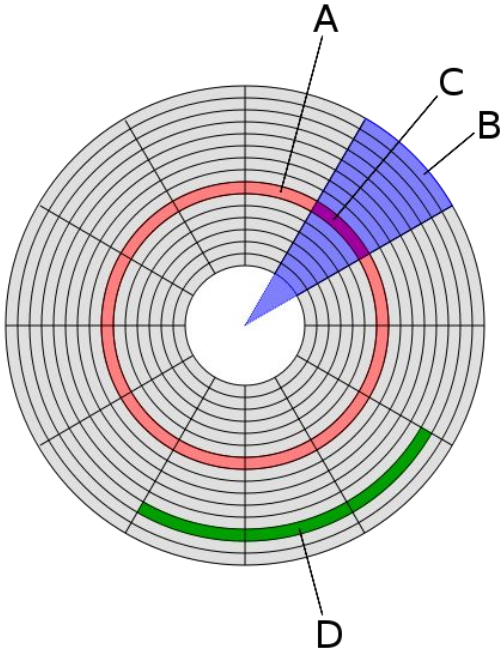
STORAGE DEVICE: INTERNALS



HARD DISK DRIVE (HDD) INTERNALS

This is one side of a single platter
A: track (<0.5M per radius inch)
B: sector (<100 per track)
C: track sector
D: a file

Spindle thru' the center
Rotates at: 7.5k to 15k rpm
R/W head for each side of platter



HDD LATENCY: SEEK, ROTATE, TRANSFER

Seek cost for the head:
- Not purely linear cost
- Accelerate, coast, decelerate, & settle
- Settling alone can take 0.5 - 2 ms

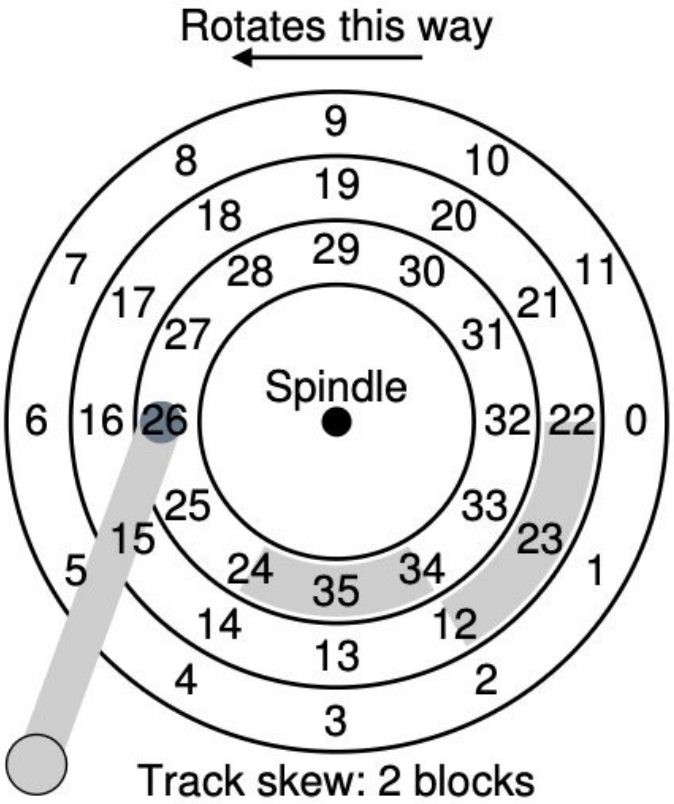
Entire seeks often takes 4 - 10 ms
- Average seek = 1/3 of max seek

Depends on rotations per minute (rpm)
- 7.2k rpm is common, 15k rpm is high end

Average rotation – half of a rotation

HDD:
100+ MB/s is typical for max transfer rate

HDD: LBA NUMBERING



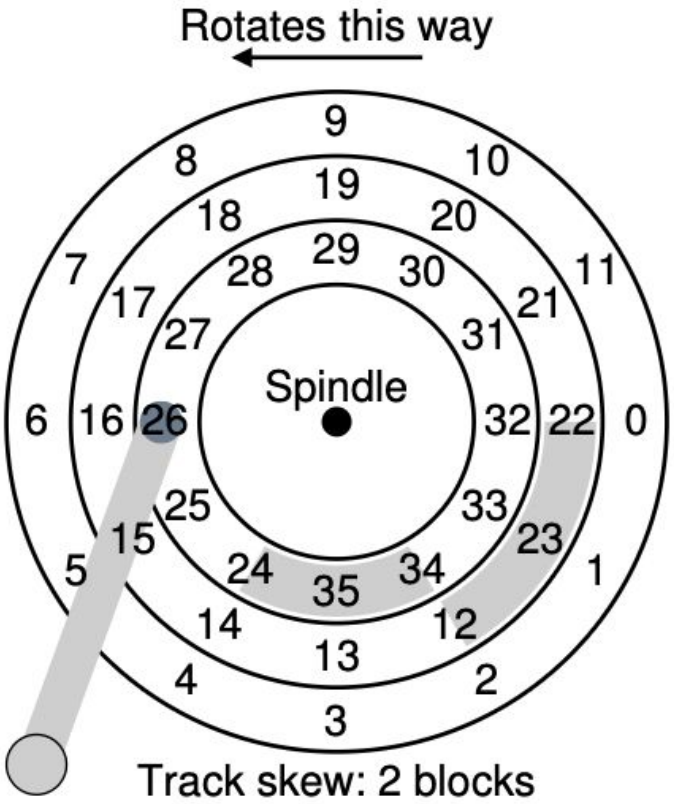
Note: Pic is misleading!

- outer tracks are as dense
- and have more LBAs

Sequential numbering

- easy to read as disk spins
- **Inter-track skew?**

HDD: LBA NUMBERING



Note: Pic is misleading!

- outer tracks are denser
- more LBAs per track

Sequential numbering

- easy to read as disk spins
- **Inter-track skew?**

seek-time * rotational speed = skew

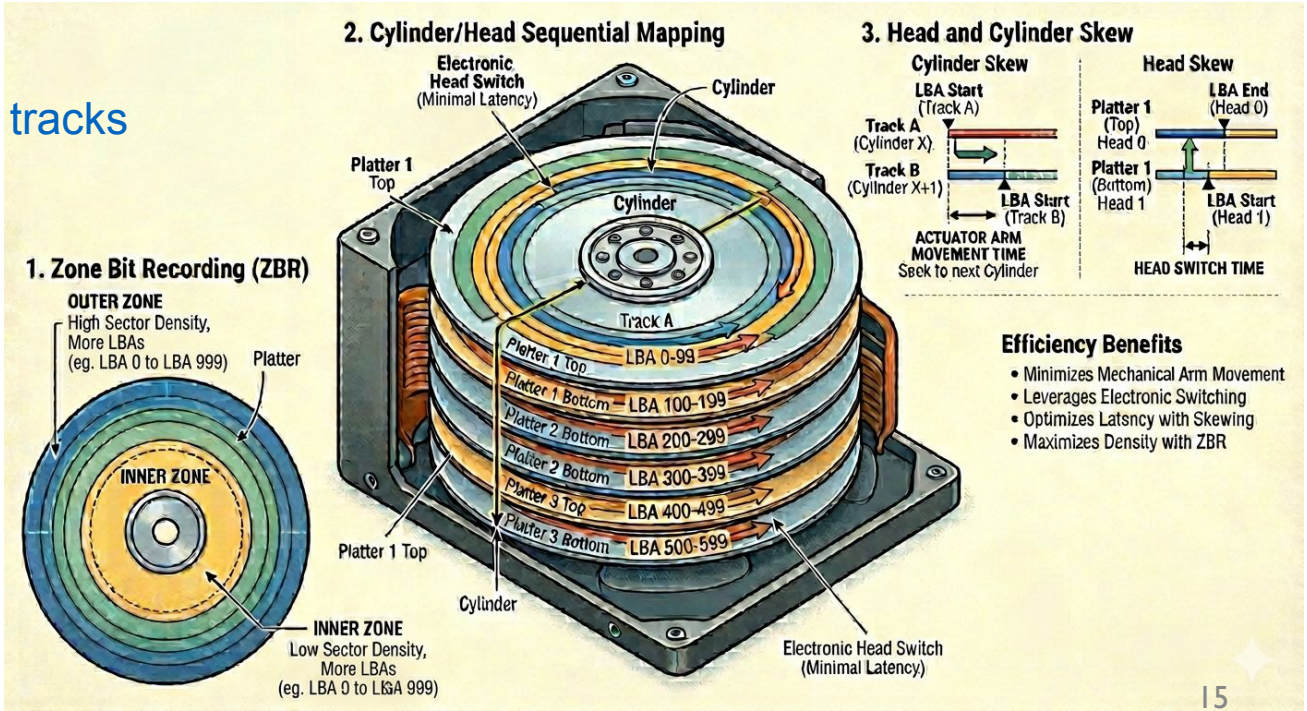
seek-time: move head to next track

HDD: LBA NUMBERING MULTI-PLATTER

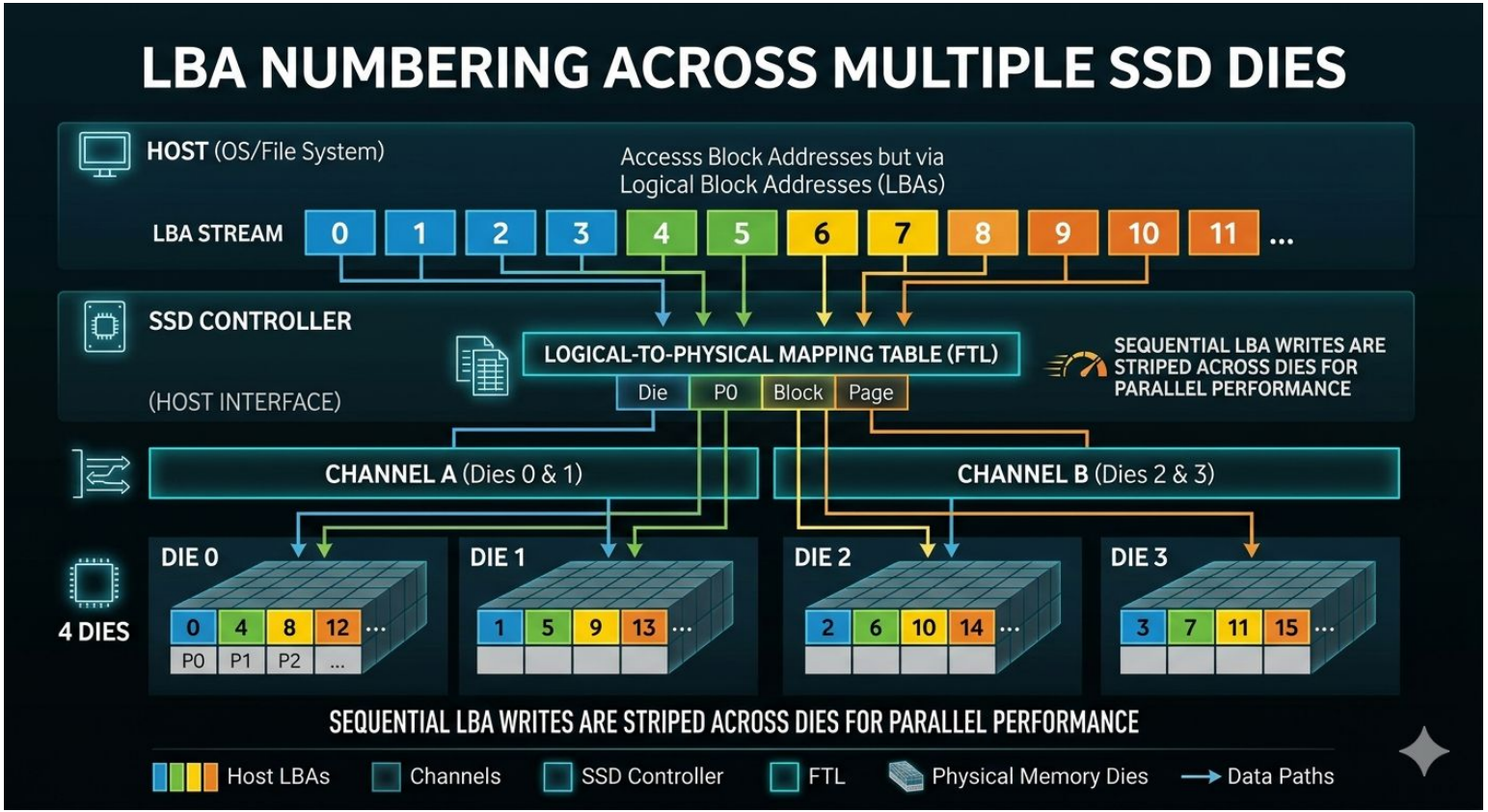
LBA numbering starts: outer-most track

- Skew for side/platter switch
  - Head reads efficiently
- Skew for every track

ZBR: fewer blocks in inner tracks



SSD: LBA NUMBERING



SSD V HDD

	HDD	SSD
Moving parts	Many parts that can fail. Heavier. Damaged by shock/vibration.	None. Faster, quieter, lighter & smaller
Power & cooling	Power to move parts; mechanical parts generate heat	SATA: no cooling needed. NVMe: heat-sink or airflow to avoid thermal throttling
Wear out	magnetic bits don't	Cannot overwrite! Erase + write Finite #writes of flash DWPD (drive writes per day) FTL: wear-leveling
Data loss when powered off	5-10 years; mechanical parts fail before magnetic data loss	1-5 years; power up at least once a year
Cost	Cheaper	Expensive

SEQUENTIAL VS RANDOM ACCESS

Sequential vs Random: LBA number space

- LBAs *i, i+1, i+2*,...are sequential

Access of *n* sequential blocks faster than *n* random blocks

- HDD: minimal head movement (only to next track)
- SDD: read in parallel across dies

Performance metrics: read vs write

- Latency: single request (ms)
- Throughput: steady state (GB/s)

I/O SCHEDULING IN DEVICE

Given a stream of R/W requests, re-order for optimal perf:

- Scheduler in the HDD/SSD firmware
- HDD: optimize for arm movement
- SSD: optimize for FTL wear-leveling

	HDD		SSD	
	Random	Sequential	Random	Sequential
Access	2-15 ms	2 ms	50-150 us	35-500 us
R Throughput	1-3 MB/s	80-160 MB/s	30-500 MB/s	500-550 MB/s
W Throughput	0.5-1 MB/s	80-160 MB/s	0.2-3 GB/s	450-500 MB/s

WHY STOP AT 1 STORAGE DEVICE?

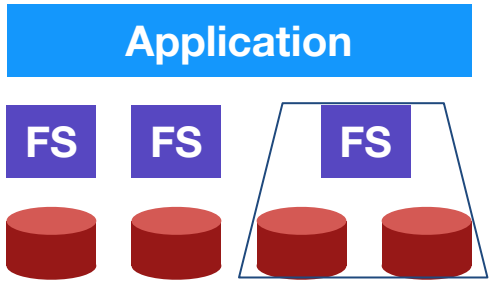
Multiple devices gives us more:

- capacity
- reliability
- performance

Challenge for OS & file systems

complexity of dealing with many SSDs/HDDs

SOLUTION 1: JBOD/JBOF



JBOD/JBOF: **J**ust a **B**unch **O**f **D**isks/**F**lash

Map each file system to 1 or few device(s)

- Simple FS-block# to LBA mapping

Application is smart

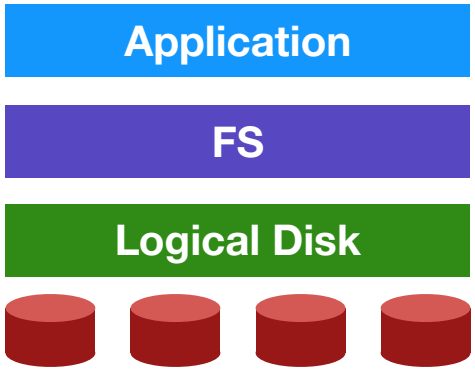
- Store different files on different file systems

Not great : unnecessary complexity for applications

SOLUTION 2: RAID

Build logical space from many physical devices:

- capacity
- reliability
- performance

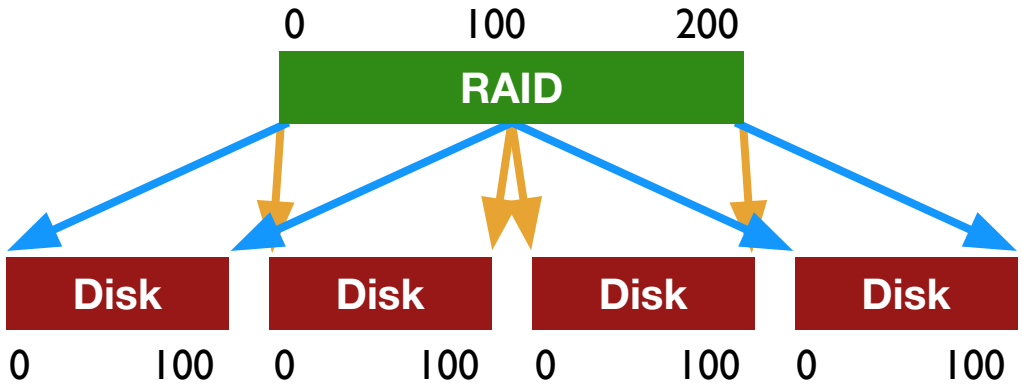


RAID: **R**edundant **A**rray of **I**nexpensive **D**isks

Transparency: no changes to the FS, Apps → ease of deployment

Seminal paper from 1988: Patterson, et al: A case for redundant arrays of inexpensive disks.

GENERAL STRATEGY: MAPPING, REDUNDANCY



Side note: Mapping “logical” block address to “physical” block address

- Mappings happen at several layers of a system
  - app to a file, file system to storage, RAID to all devices, a disk (controller) to the physical storage space
- Mapping can be **Dynamic** (hash table, tree) or **Static** (use simple math)

ANALYSIS OF RAID

Assume “fail stop” model: a disk either works or not

- Once a disk stops working, it stays that way
- Working-or-not status is detectable

Workloads: latency (single request) & throughput (steady state)

- Random vs Sequential & Reads vs Writes
  - N: #disks
  - C: capacity of 1 disk
  - S/R: sequential/random throughput of 1 disk
  - D: latency of one small I/O operation

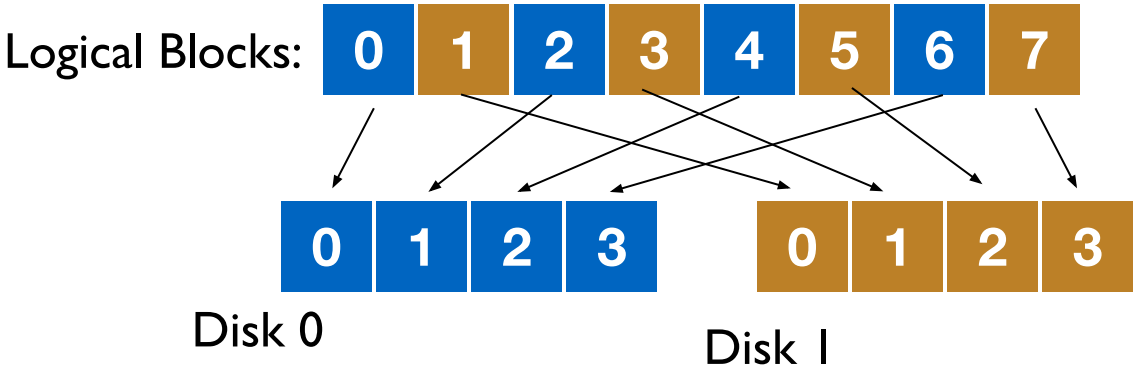
Metrics:

- Capacity: Total storage space available
- Reliability: How many failed disks can be tolerated?
- Performance: latency & throughput

Trade-offs made by each RAID level

RAID-0

Optimize for capacity. No redundancy



RAID 0: STRIPES AND CHUNK SIZE

	Disk 0	Disk 1	Disk 2	Disk 4
	0	1	2	3
stripe:	4	5	6	7
	8	9	10	11
	12	13	14	15

Chunk size = 1block

assume chunk size of 1 for this lecture

	Disk 0	Disk 1	Disk 2	Disk 4
	0 1	2 3	4 5	6 7
stripe:	8 9	10 11	12 13	14 15

Chunk size = 2 blocks

Stripe: blocks/chunks on each disk at the same LBA

RAID-0: ANALYSIS

Capacity?

Reliability: how many failed disks can be tolerated?

Latency (random):

Throughput (sequential, random):

N: #disks
C: capacity of 1 disk
S/R: sequential/random throughput of 1 disk
D: latency of one small I/O operation

RAID-0: ANALYSIS

Capacity?

N * C

Reliability: how many failed disks can be tolerated?

0

Latency (random):

D

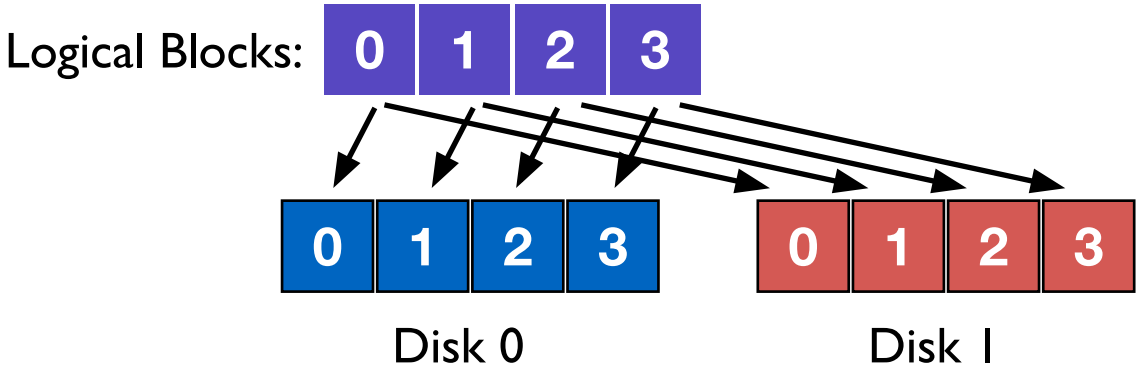
Throughput (sequential, random):

N*S, N*R

More disks improves throughput not latency

N: #disks
C: capacity of 1 disk
S/R: sequential/random throughput of 1 disk
D: latency of one small I/O operation

RAID-1: MIRRORING



Store 2 copies of each block: one in each disk