CS 370: Operating Systems
[Mass Storage]

Shrideep Pallickara
Computer Science
Colorado State University

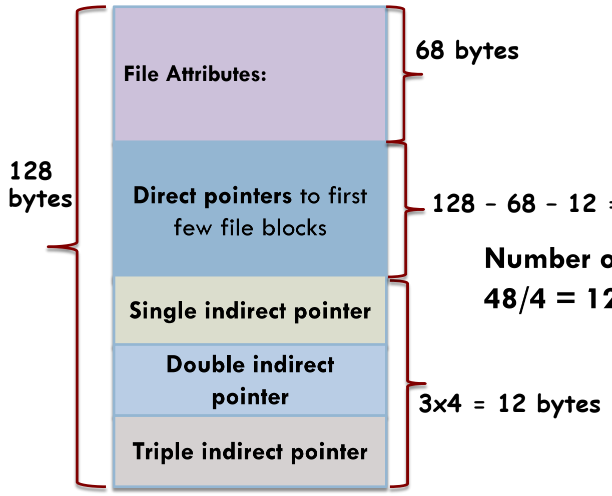
Frequently asked questions from the previous class survey

- How does NTFS compare with UFS?

Topics covered in this lecture

- Wrap-up of iNodes
- Flash Memory
- RAID
- Final Exam

inode: A quantitative look
BLOCK Size = 8 KB and Pointers = 4 bytes

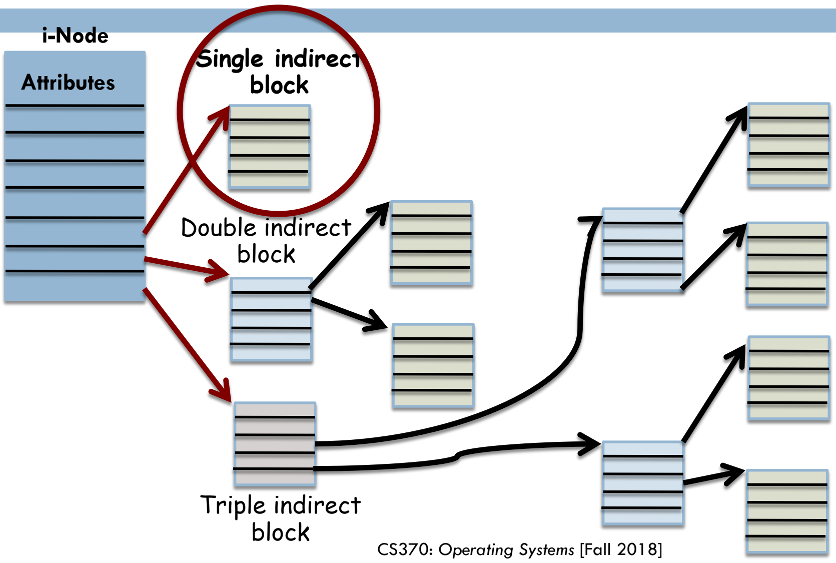


Number of direct pointers?
48/4 = 12

inode: A quantitative look
BLOCK Size = 8 KB and Pointers = 4 bytes

- 12 **direct** pointers to file blocks
- Each file block = 8KB
- Size of file that can be represented with direct pointers
  - 12 x 8 KB = 96 KB

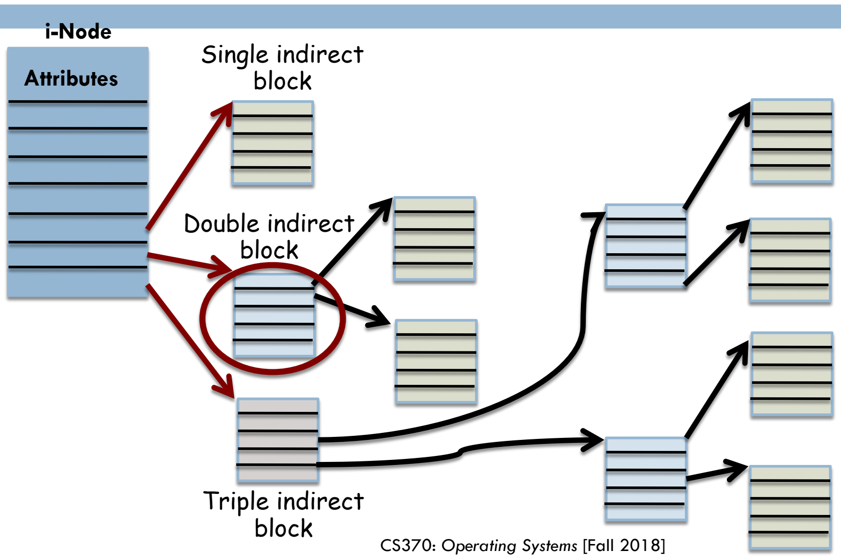
inode



inode: A quantitative look
BLOCK Size = 8 KB and Pointers = 4 bytes

- Block size = 8 KB
- Single indirect block = block size = 8 KB (8192 bytes)
- Number of pointers held in a single-indirect-block
  - Block-size/Pointer-size
  - 8192/4 = 2048
- With single-indirect pointer
  - Additional 2048 x 8 KB = 2^{11} x 2^{3} x 2^{10} = 2^{24} (16 MB) of a file can be addressed

inode



inode: A quantitative look
BLOCK Size = 8 KB and Pointers = 4 bytes

- With a **double indirect pointer** in the inode
  - The double-indirect block has 2048 pointers
    - Each pointer points to a different single-indirect-block
    - So, there are 2048 single-indirect blocks
  - Each single-indirect block has 2048 pointers to file blocks
- Double indirect addressing allows the file to have an additional size of
  - $2048 \times 2048 \times 8\ \text{KB} = 2^{11} \times 2^{24} = 2^{35}$ …. (32 GB)

inode: A quantitative look
BLOCK Size = 8 KB and Pointers = 4 bytes

- **Triple indirect addressing**
  - Triple indirect block points to 2048 double indirect blocks
  - Each double indirect block points to 2048 single indirect block
  - Each single direct block points to 2048 file blocks
- Allows the file to have an additional size of
  - $2048 \times 2048 \times 2048 \times 8\ \text{KB} = 2^{11} \times 2^{35} = 2^{46}$ (64 TB)

Limits of triple indirect addressing

- In our example:
  - There can be 2048 x 2048 x 2048 data blocks
  - *i.e.*, $2^{11} \times 2^{11} \times 2^{11} = 2^{33}$
  - Pointers would need to be longer than 32-bits to fully address this storage

What if we increase the size of the pointers to 64-bits (data block is still 8 KB) ?

- What is the maximum size of the file that we can hold?
- 8 KB data block can hold (8192/8) = 1024 pointers
- **Single indirect** can add
  - $1024 \times 8\text{ KB} = 2^{10} \times 2^{3} \times 2^{10} = 2^{23}$ (8MB) of additional bytes to the file

What if we increase the size of the pointers to 64-bits (data block is still 8 KB)?

- **Double indirect** addressing allows the file to have an additional size of
  - 1024 x 1024 x 8 KB = 2^{10} x 2^{23}= 2^{33} …. (8 GB)
- **Triple indirect** addressing allows the file to have an additional size of
  - 1024 x 1024 x 1024 x 8 KB = 2^{10} x 2^{33}= 2^{43} (8 TB)

FLASH MEMORY

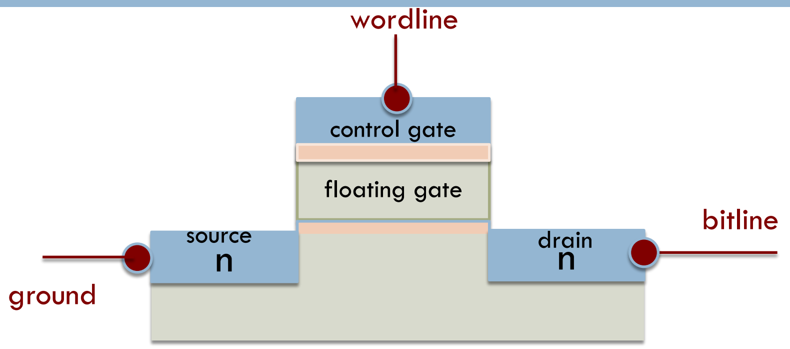
Flash memory is a type of a solid state storage

- No moving parts … and stores data using electrical circuits
  - Can have better random I/O performance than HDDs, use less power, and is less vulnerable to physical damage
  - But significantly more expensive per byte

Transistors

- It takes one transistor to store a bit
- Ordinary transistors are electronic switches
  - Turned on and off by electricity
- Strength: Computer can store information simply by **passing patterns of electricity** through its memory circuits
- Weakness: As soon as power is turned off, transistors revert to their original state (loses all information)
  - Electronic amnesia!

Transistors in flash memory



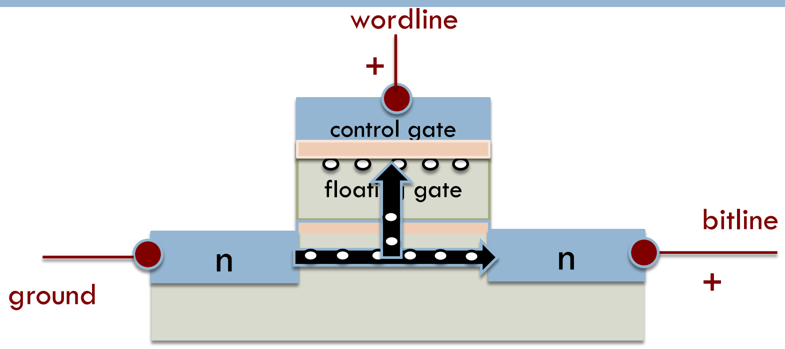
The source and drain regions are rich in electrons (n-type silicon)

Electrons cannot flow from source to drain, because of the electron-deficient p-type material between them

A gate that floats?

- The extra gate in our transistor "floats" — it is not connected to any circuit
- Since the floating gate is entirely surrounded by an **insulator**, it will hold an electrical charge for months or years without requiring any power
- Even though the floating gate is not electrically connected to anything, it can be charged or discharged
  - Via **electron tunneling** by running a sufficiently high-voltage current near it

Transistors in flash memory



The presence of electrons on the floating gate is how a flash transistor stores a **one**

Electrons stay there indefinitely, even when positive voltages are removed AND whether power is supplied to the unit or not

Electrons can be flushed out by putting a negative voltage on the wordline. REPELS electrons back.

Flash storage: Erasure blocks

- Before flash memory can be written, it must be *erased* by setting each cell to a logical "1"
- Can only be erased in large units called **erasure blocks** (128-512 KB)
- Slow operation: takes several milliseconds
- Erasing an erasure block is what gives "flash memory" its name …
  - Resemblance to the flash of a camera

Write page and read page

- Write Page:
  - Once erased, flash memory can be written on a page-by-page basis
  - Each page is typically 2-4 KB
  - Writing a page takes about 10s of microseconds
- Read page
  - Flash memory is read on a page-by-page basis
  - Reading a page takes about 10s of microseconds

Challenges in writing to a page

- To write a page, it's **entire erasure block** must first be erased
  - Erasure is slow and affects a large number of pages
- Flash translation layer (FTL)
  - Maps logical flash pages to different physical pages on the flash device
  - When logical page is overwritten, the FTL writes the new version to a free, already-erased physical page
    - … and remaps logical page to that physical page
  - Write remapping significantly improves performance

Durability [1/2]

- Normally, flash memory can retain state for months or years without power
- However, **high current loads** from flashing and writing memory *causes circuits to degrade*
  - After a few 1000~1,000,000 erase cycles a cell may wear out … cannot reliably store a bit

Durability [2/2]

- Reading a flash memory cell a large number of times causes surrounding cells' charges to be **disturbed**
  - **Read disturb error**: Location in memory read too many times without surrounding memory being rewritten

Improving durability

- Error correcting codes
- Bad page and bad erasure block management
  - Firmware stops storing data on defective blocks
- **Wear leveling**
  - Move logical pages to different physical pages to ensure *no physical page gets inordinate number of writes* and wears out prematurely
  - Some algorithms also migrate unmodified pages to protect against read disturb errors
- Spare pages and erasure blocks

Parameters for the Intel 710 Series SSD

- Capacity 300 GB
- Page Size 4 KB

- Performance
  - Bandwidth (Sequential Reads) 270 MB/s
  - Bandwidth (Sequential Writes) 210 MB/s
  - Read/ Write Latency 75 μs
  - Random Reads Per Second 38,500
  - Random Writes Per Second 2,000
    - 2,400 with 20% space reserve

Parameter for the Intel 710 Series SSD

- Interface SATA 3 Gb/ s
- Endurance
  - Endurance 1.1 PB
  - 1.5 PB with 20% space reserve
- Power
  - Power Consumption Active/ Idle 3.7 W / 0.7 W

RAID STRUCTURE

RAID involves using large number of disks in parallel

- Improves **rate** at which data be can read/written
- Increases **reliability** of storage
  - Redundant information can be stored on multiple disks
  - Failure of 1 disk should not result in loss of data
- Redundant Array of ~~Inexpensive~~ **Independent** Disks

RAID levels

- Standardized by the Storage Networking Industry Association (SNIA)
  - In the Common RAID Disk Drive Format (DDF) standard
- Originally there were 5 levels
- There are other nested levels

Reliability through redundancy

- Store information that is not normally needed
- Can be used in the event of disk failure
  - **Rebuild** lost information
- Simplest approach: **Mirroring**
  - Duplicate every disk
  - Data lost only if 2nd disk fails BEFORE 1st one is replaced
  - Watch for: Correlated failures

RAID parallelism

- **Stripe** data across disks
- Objectives
  1. Increase throughput
  2. Reduce response times of large accesses

RAID Parallelism: Stripe data across disks Bit level striping

- **Split bits** of each byte across multiple disks
  - 8 disks: Bit **`i`** of each byte written to disk **`i`**
  - Bit 3 written to disk 3
- Array of 8 disks treated as a single disk
  - 8 times the access rate
  - Every disk participates in every read/write

RAID Parallelism: Block-level striping

- **Blocks** of a file are striped across multiple disks
- When there are **`n`** disks
  - Block **`i`** of the file written to …
  - Disk: **`(i mod n) + 1`**
    - 4 disks: Block 9 of file goes to disk 2
    - 4 disks: Block 8 of file goes to disk 1

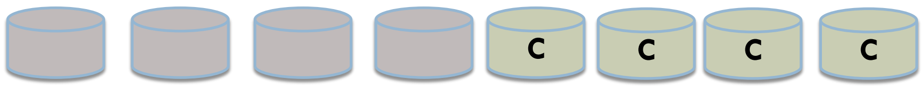
RAID levels

- Striping improves transfer rates
  - BUT not reliability
- Disk striping usually combined with **parity**
- Different schemes classified according to levels
  - RAID levels

RAID 0: Stripe blocks without redundancy

- No mirroring
- No parity

RAID 1: Disk mirroring

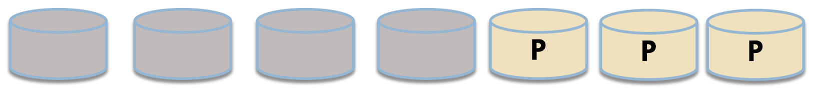


- Each disk is mirrored

RAID 2: Memory style error correcting code

- **Parity bit** records number of 1 bits in byte
  - Even: parity 0
  - Odd: parity 1
- Use to detect single-bit errors
- Error correcting schemes
  - 2 or more extra bits to recover from single-bit errors

RAID 2: Error Correcting Codes



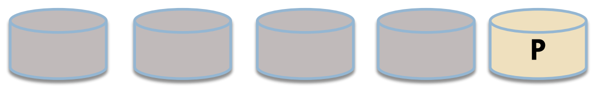
Error correction bits

- **If one disk fails:**
- Remaining bits of the byte **+** error correction bits
  - Read from other disks
- **Reconstruct** damaged data

RAID 3:
Single parity bit used for error correction

- We can identify damaged sector
- Figure out if any bit in sector is 0 or 1
  - Compute parity of corresponding bits from other sectors
    - If parity of remaining bits == stored parity
      - Missing bit = 0
    - Otherwise, missing bit = 1

RAID 3: Single parity bit used for error correction



Error correction bits

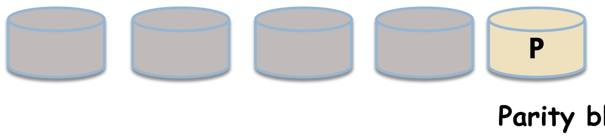
Issues

- Fewer I/Os per-second since every disk participates in every I/O
- Overheads for computing parity bits

RAID-4 Block interleaved parity

- Block-level striping
  - Block read accesses only one disk
  - Data transfer rate slower for each access
  - Multiple reads proceed in parallel
    - Higher overall I/O rate
- **Parity block** on a separate disk

RAID 4:
Block interleaved parity



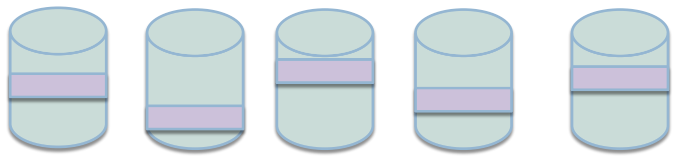
If one disk fails

- Parity block used with corresponding blocks
  - Restore blocks of failed disk

RAID-5
Block interleaved distributed parity

- **Spread** data and parity among all **`N+1`** disks
  - Avoid overuse of single parity disk
- Parity block does not store parity for blocks on the same disk

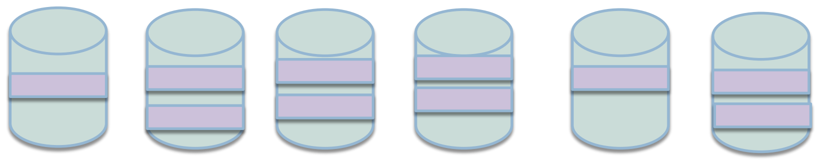
RAID 5: Block interleaved distributed parity



RAID-6

- Store extra redundant information
  - Guard against **multiple** disk failures
- Error correcting codes are used
  - Reed-Solomon codes
- 2-bits of redundant data
  - For every 4-bits of data

RAID-6



In the computer science department

- RAID 1
  - To mirror the root disks of the servers
- RAID 5
  - For all the "no_back_up" partitions
- RAID 6
  - For all data disks

FINAL EXAM

Final exam

- In our regular classroom (Clark A-103) from
  - There will be NO MAKEUP exam
- Comprehensive exam
  - Covers **ALL** topics that we have discussed in the course
- Duration: 2 hours

Breakdown

Processes and IPC	5
Threads:	10
CPU Scheduling	10
Process Synchronization & Atomic Transactions	15
Deadlocks	10
Memory Management	10
Virtual Memory	10
Virtualization	10
File Systems	15
Mass Storage & Disk Scheduling	5

The contents of this slide-set are based on the following references

- *Avi Silberschatz, Peter Galvin, Greg Gagne. Operating Systems Concepts, 9th edition. John Wiley & Sons, Inc. ISBN-13: 978-1118063330.* [Chapter 10, 11]
- *Andrew S Tanenbaum and Herbet Bos. Modern Operating Systems. 4th Edition, 2014. Prentice Hall. ISBN: 013359162X/ 978-0133591620. [Chapter 5]*
- *Thomas Anderson and Michael Dahlin. Operating Systems: Principles & Practice. 2nd edition. ISBN: 978-0-9856735-2-9 [Chapter 12]*
- *Chris Woodford. Flash Memory. http://www.explainthatstuff.com/flashmemory.html*