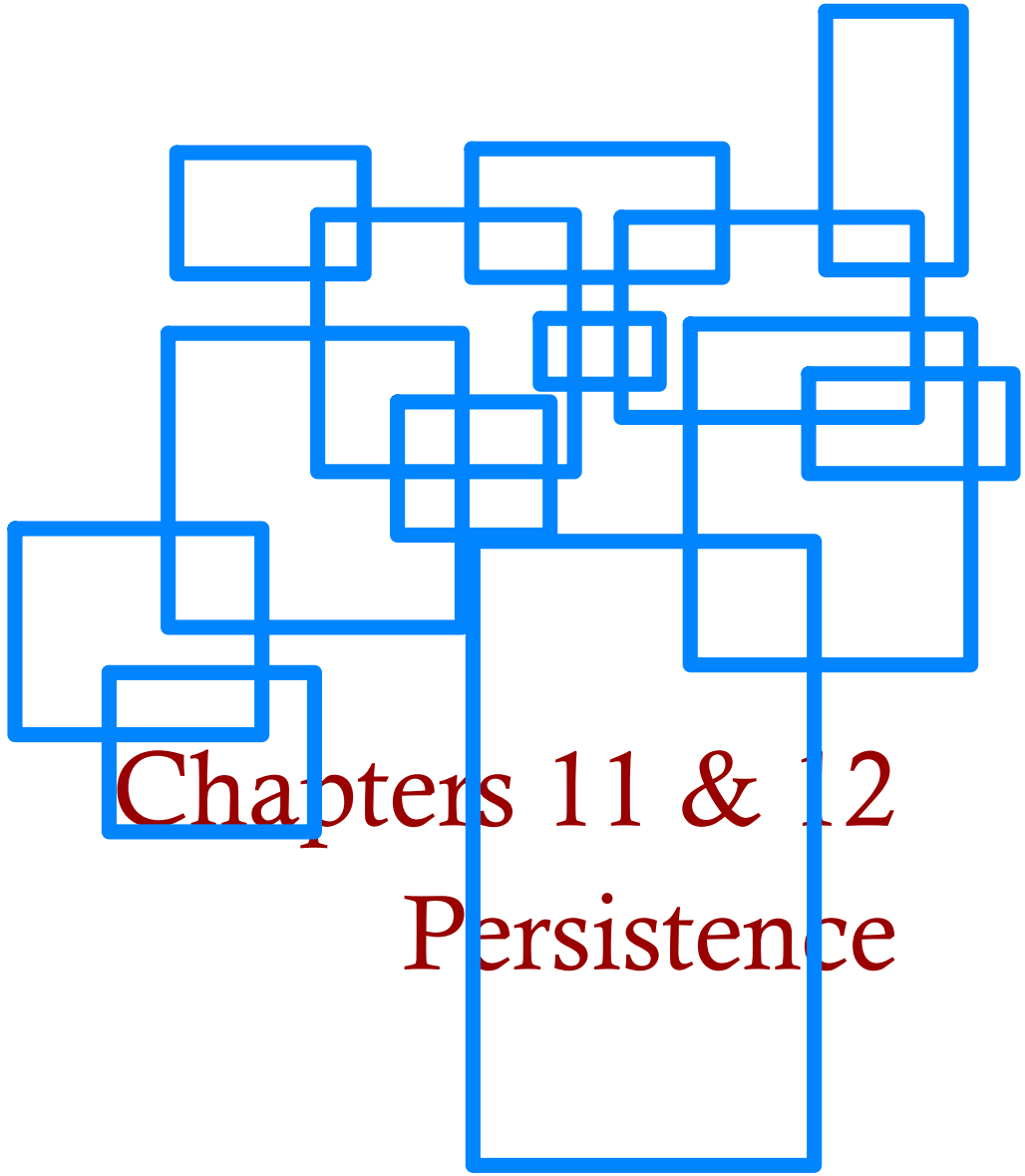


Chapters 11 & 12
Persistence

Part 0: Introduction

Persistent Storage

a device which stores information permanently, even through a power loss

→ Disk, SSDs, magnetic tape
↳ hard drive

→ Not RAM

Crux of the problem: How does the OS manage a persistent device?

Files and Directories

File: a linear array of bytes each of which you can r or w

→ a low level name (inode number)

Directory: a list of tuples: (user readable name, inode number)

- also have an inode number
- so we can nest directories → directory tree
- The root directory: "/" in Unix

Example Directory Tree

need a separator:
"/" - sep in Unix

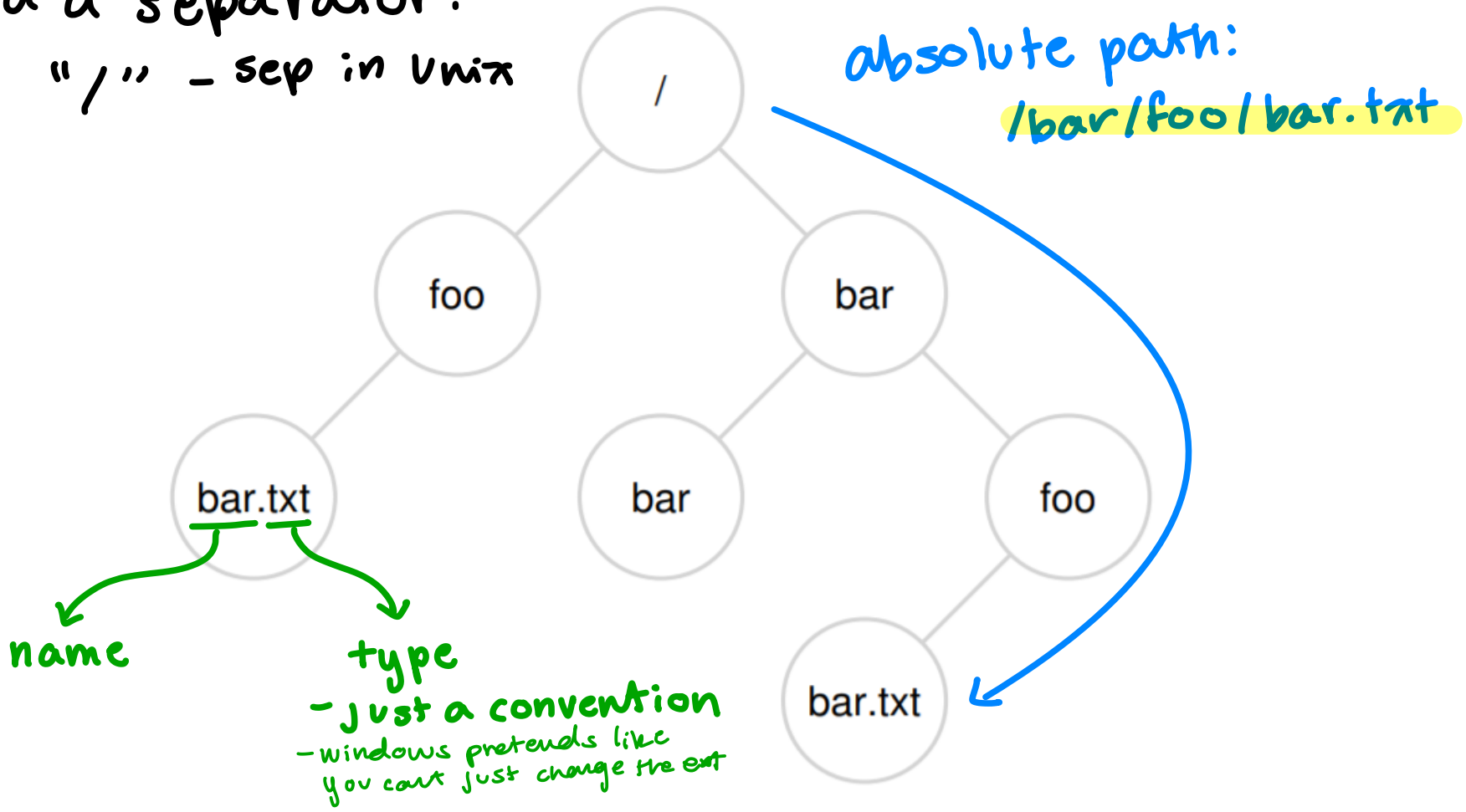


Figure 39.1: **An Example Directory Tree**

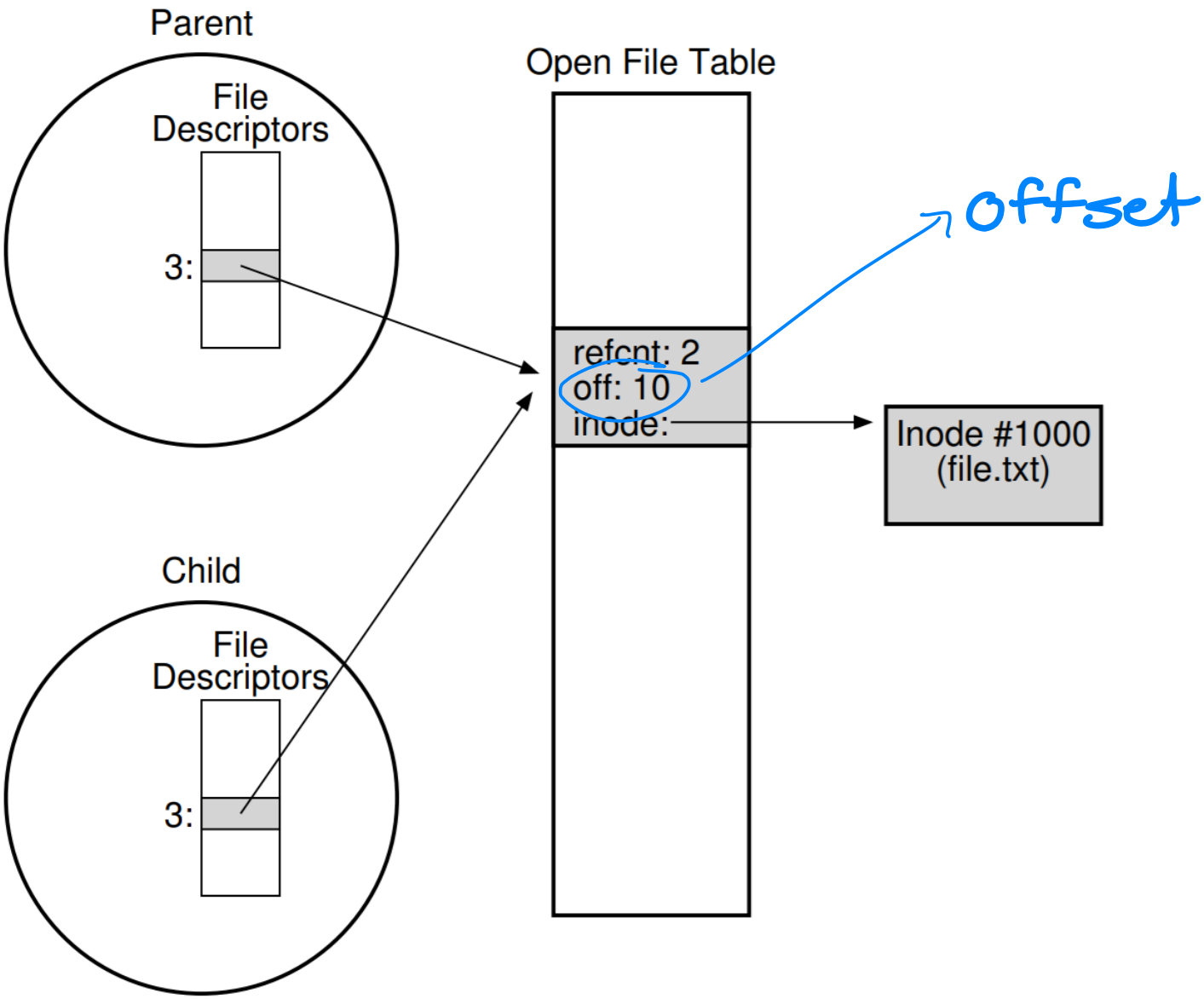
(Unix) File System Interface

- `open()`
- `read()`
- `write()`
- `close()`
- `dup()`
- `dup2()`

"dup() and friends"

→ file descriptors ≠ inode number

File Descriptor => Inode



File System Implementations

In general:

- Pure software
- lots of flexibility
- lots of different solutions

⟹ device characteristicss affect performance greatly

Part 1: Disk

Setup

Disk is composed of *sectors*

- 512-byte block, label 0:n-1 (n sectors)
- disk is not Byte addressable - sector addr.
- Sector writes are atomic

The "unwritten contract": accessing 2 sectors near ela is faster is faster than 2 sectors far apart

★ Disk is terrible @ Random access

Geometry of Disk

Platter: a hard circular surface on which data is stored

Surface: side of a platter (2: front/back)

Track: a concentric circle of sectors, on a surface ≈ 300k tracks per surface

Spindle: bounds platters together, and is connected to a motor which spins @ a const rate (rpm)

Disk Head: a mechanism to r/w the magnetic patterns on a track → data

Visual

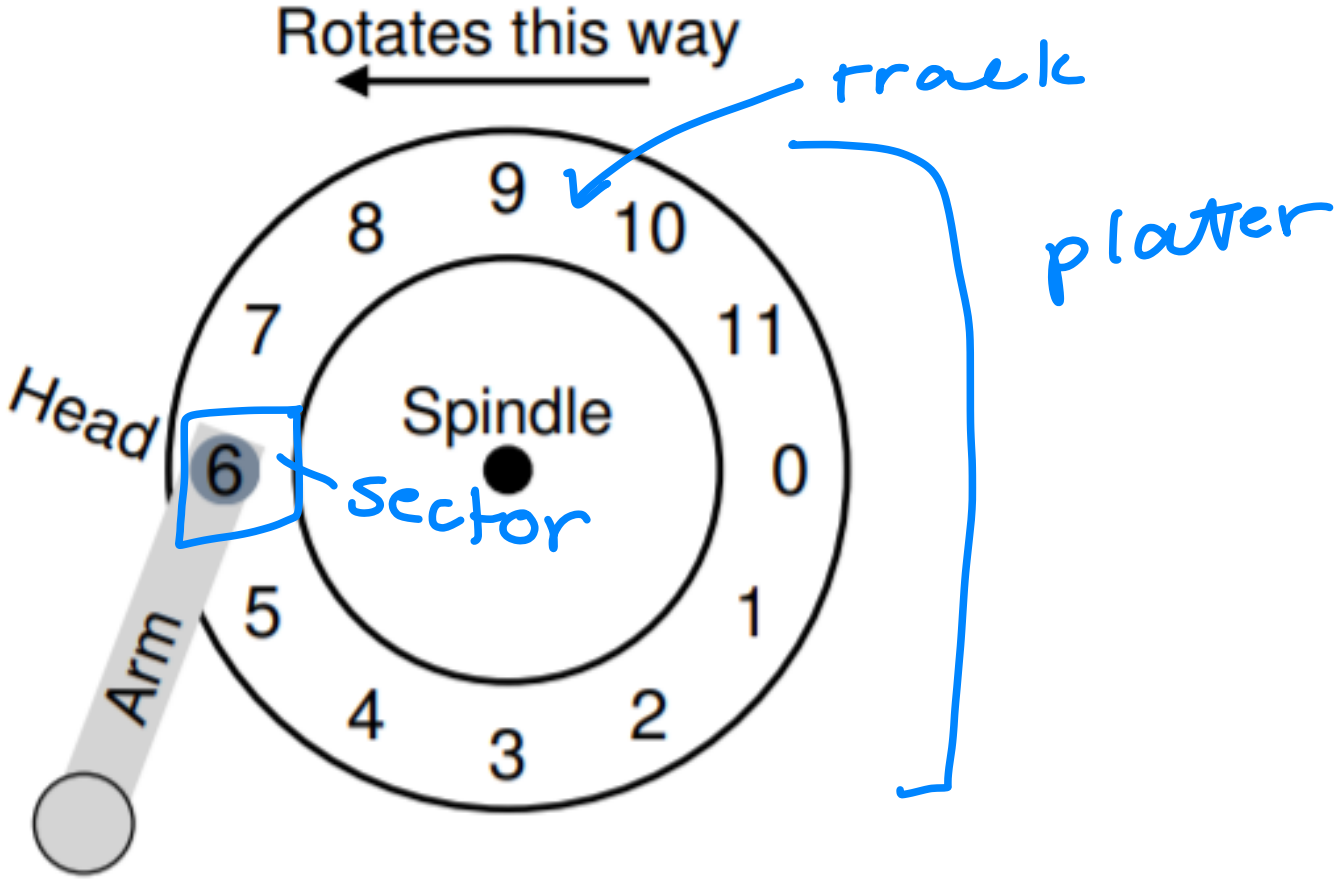


Figure 37.2: **A Single Track Plus A Head**

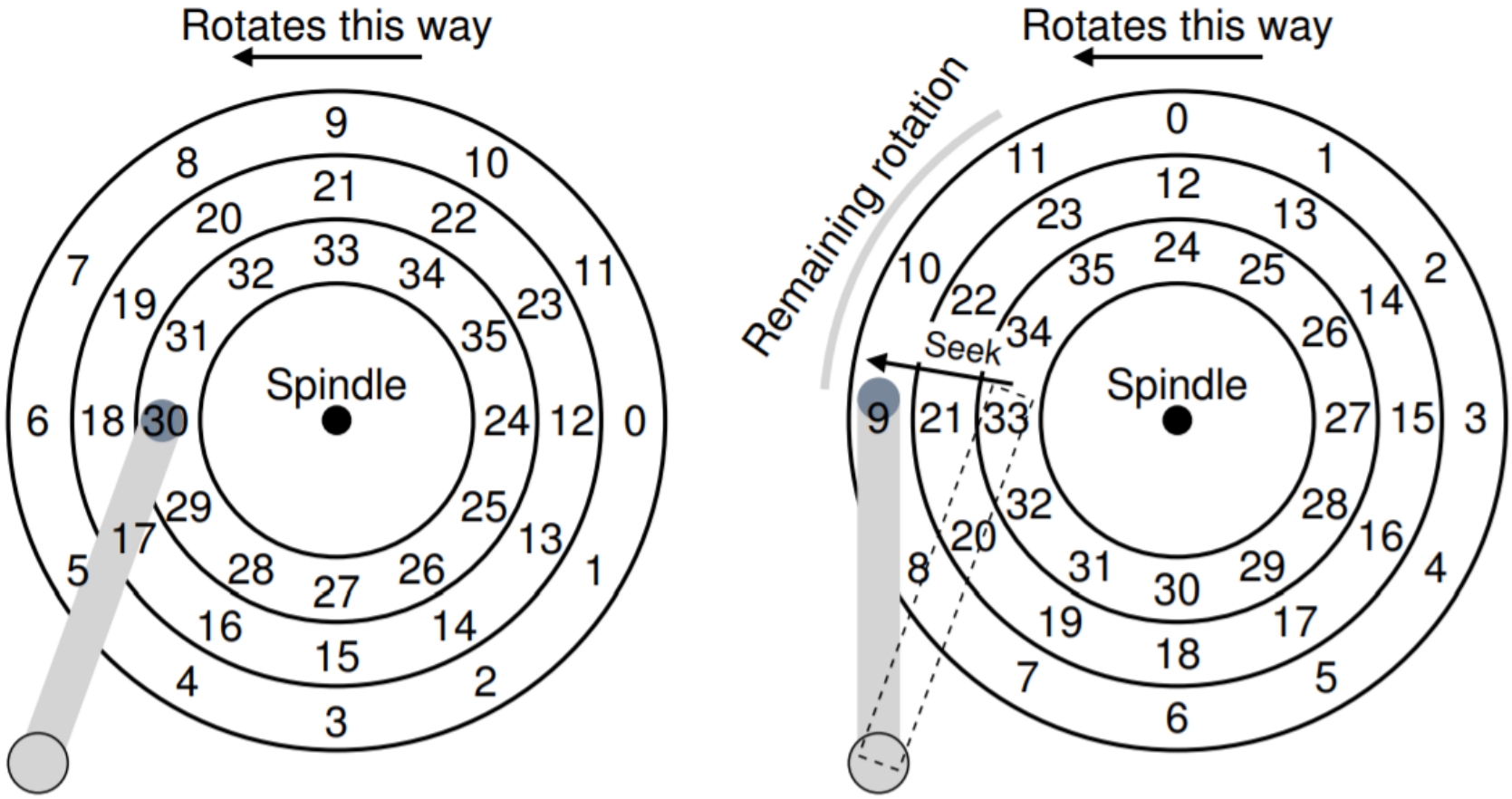
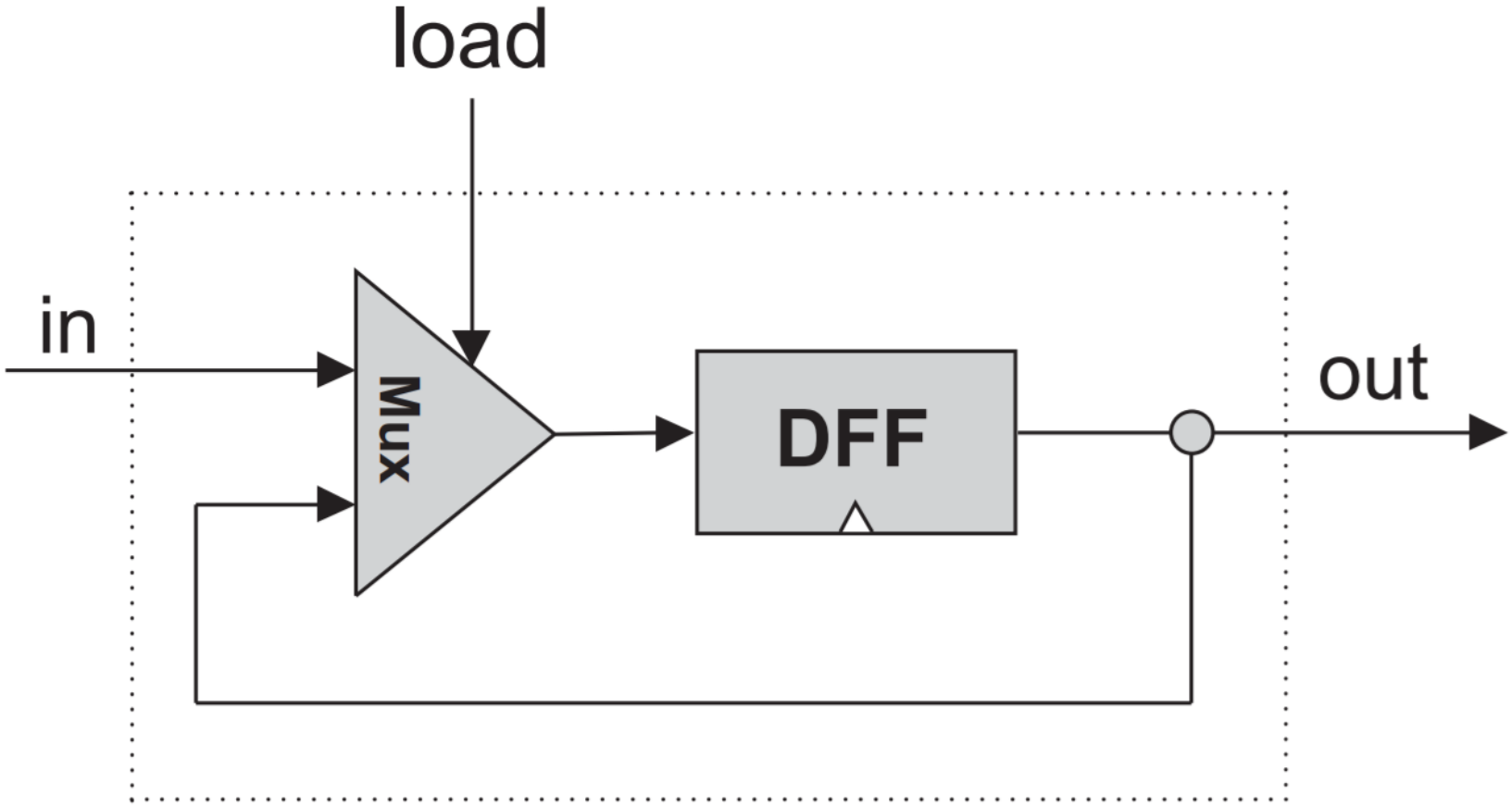


Figure 37.3: **Three Tracks Plus A Head (Right: With Seek)**

1. "seek": move head to correct track ~ms
2. "rotational delay": wait for disk to spin to the correct position
3. "transfer"

Comparison to RAM



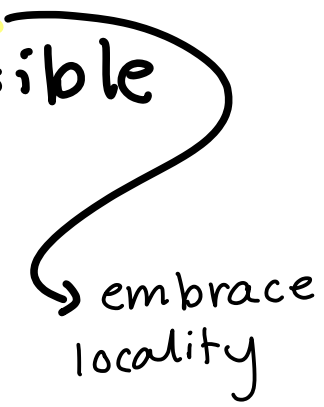
I/O Time

$$T_{I/O} = T_{seek} + T_{rotation} + T_{transfer}$$

how to reduce?

T_{seek}: move tracks as little as possible ★

T_{rot}: spin faster (not feasible)
anticipate rotation



T_{Trans}: improve physics