

I File System



Le memorie di massa

- ▶ Forse la *periferica* più importante di un elaboratore
- ▶ File system:
 - Un insieme di funzionalità per astrarre i dati grezzi presenti in memoria di massa e interpretare questi ultimi in termini di *files* e *cartelle*
 - Principalmente concepiti per gestire i dischi fissi
 - *Altre memorie di massa importanti: SSD, CD, DVD...*



Dalle applicazioni alle mmassa

Applicazioni

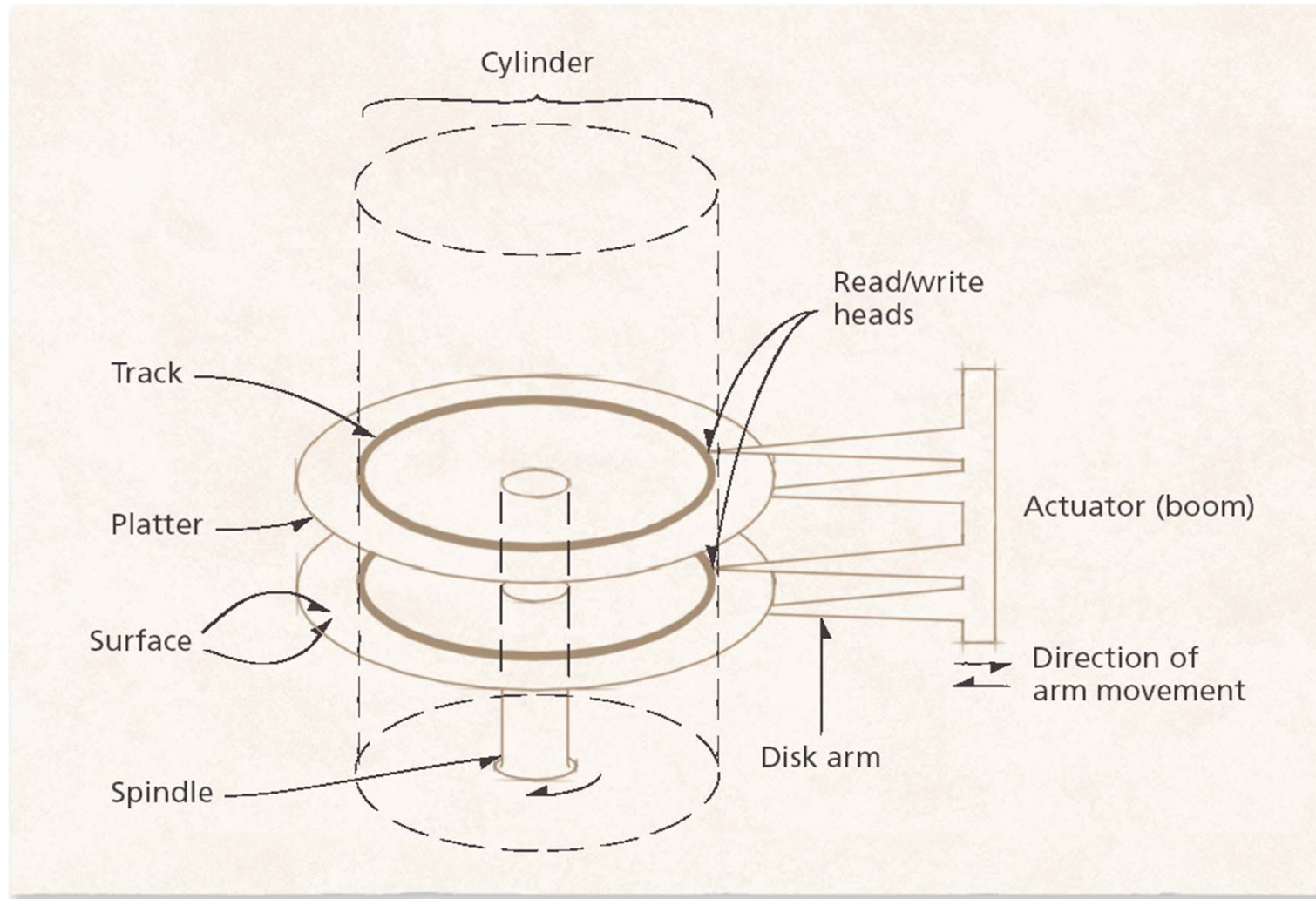
File system

HAL: Driver disco

Interfaccia (SATA, SAS, SCSI...). Numerazione LBA

PHY
(Mezzo fisico. Numerazione C/H/S)

Come è fatto un Hard disk



Parametri prestazionali

- Latenza di rotazione (in ms)
 - Tempo che ci mettono i dati richiesti ad arrivare sotto la testina (in media $1 / (\text{RPM}/60 * 2)$)
- Seek time (in ms)
 - Tempo che ci mettono le testine ad arrivare su un dato cilindro (dipende da fattori costruttivi)
- Tempo di trasmissione (in bit/sec)
 - Tempo che ci mettono tutti i dati richiesti (sequenze di bit adiacenti) ad essere letti = $\text{RPM}/60 * \text{bit per cilindro}$.



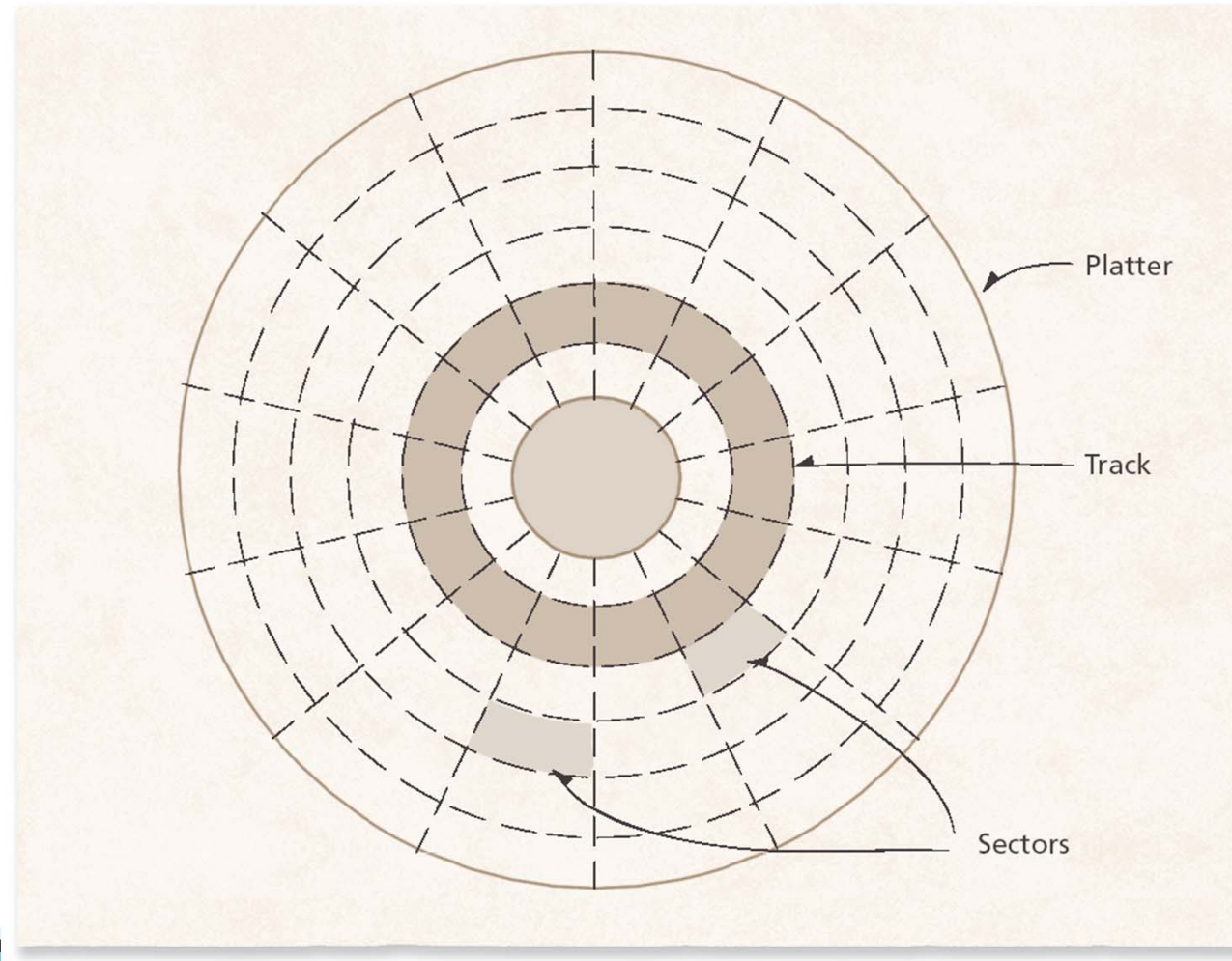
Alcuni valori plausibili

<i>Model (Environment)</i>	<i>Average Seek Time (ms)</i>	<i>Average Rotational latency (ms)</i>
Maxtor DiamondMax Plus 9 (High-end desktop)	9.3	4.2
WD Caviar (High-end desktop)	8.9	4.2
Toshiba MK8025GAS (Laptop)	12.0	7.14
WD Raptor (Enterprise)	5.2	2.99
Cheetah 15K.3 (Enterprise)	3.6	2.0

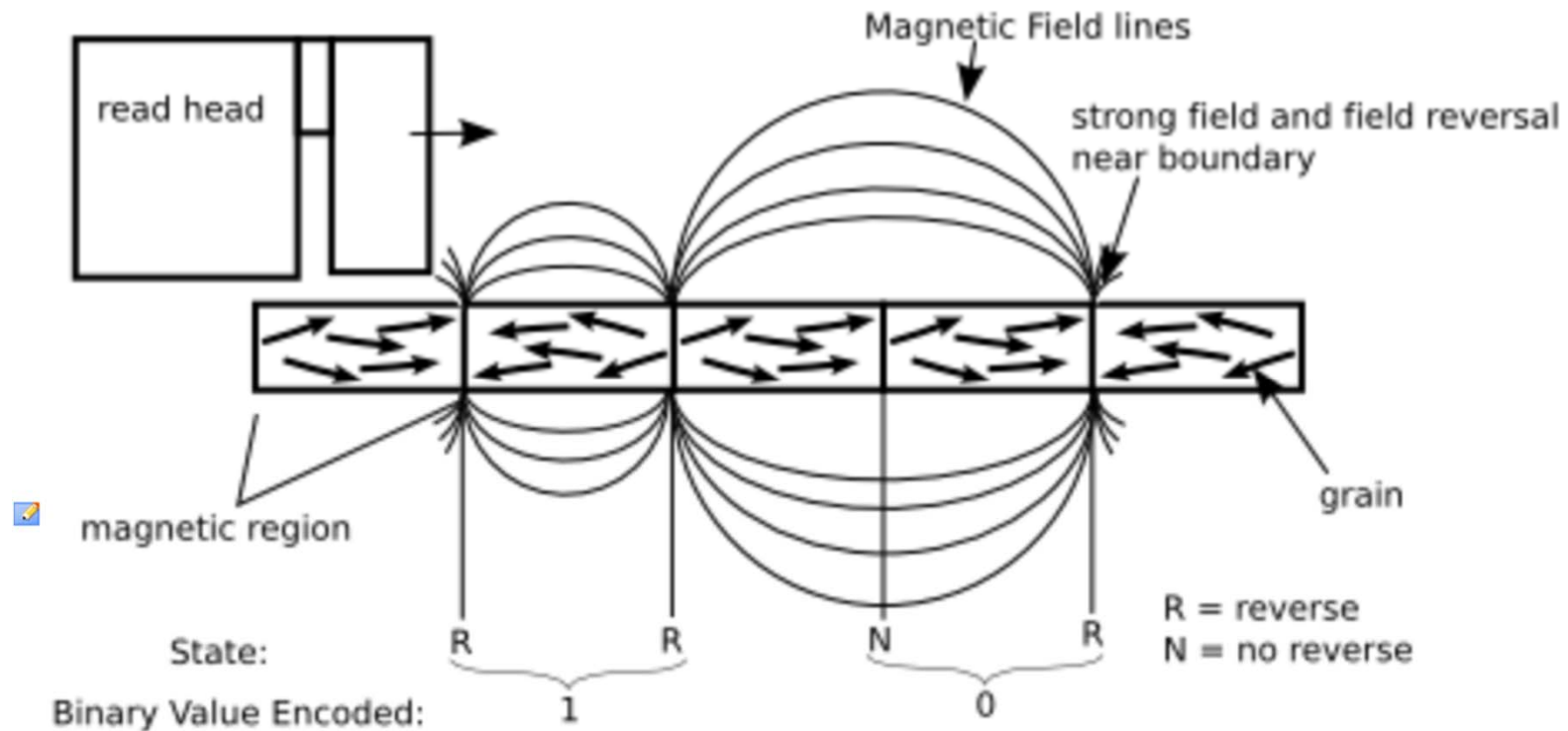


Cosa c'è su un singolo piattello

- Ogni traccia è usualmente divisa in settori di 512 byte



Memorizzazione magnetica



Che comandi posso dare a un HD?

Comandi di basso livello:

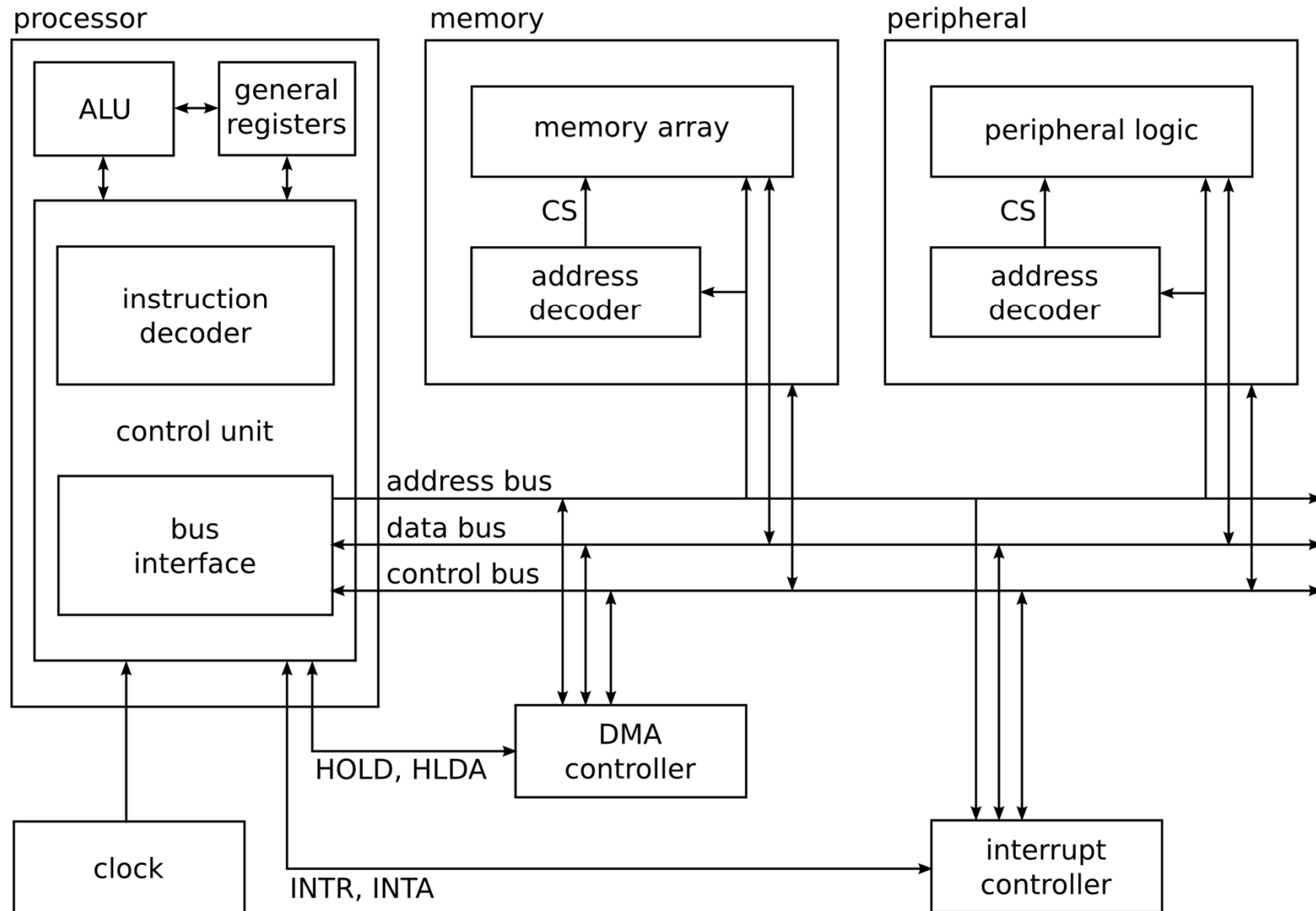
- Comandi di lettura
 - (es. READ Sector, Num → READ C/H/S)
- Comandi di scrittura
 - (es. WRITE Sector, Num → WRITE C/H/S)
- Si tratta di comandi di basso livello che consentono di leggere/scrivere 1 o più settori per volta
- Su questo livello di astrazione non esiste il concetto di file, directory, etc. etc.
- Piano di numerazione interno dei settori a 3 coordinate:
 - Cilindro (Cylinder), Testina (Head), Settore (Sector)
 - Coordinate CHS



Con che modalità i comandi vengono impartiti?

- ▶ Tecnica mista DMA/Interrupt



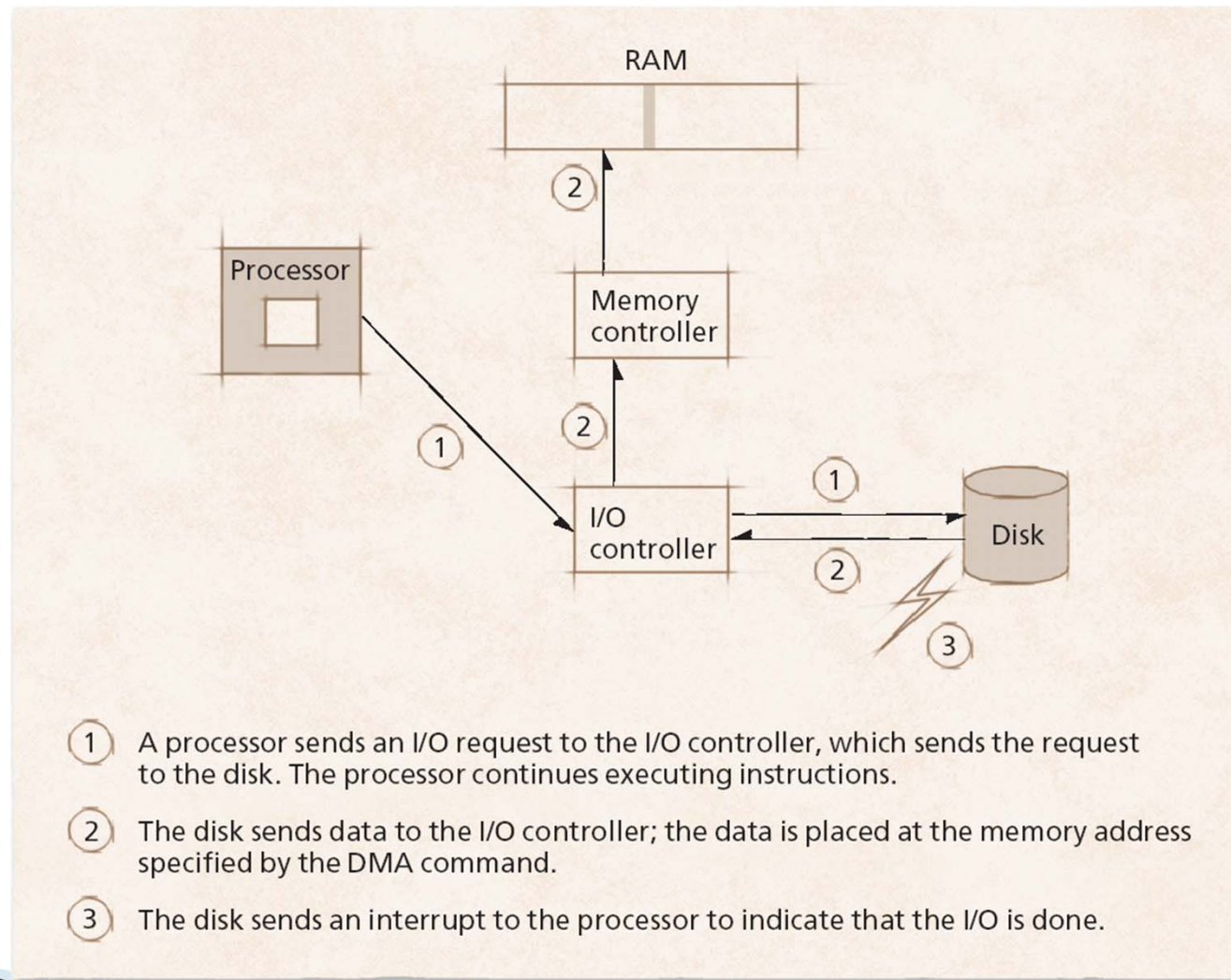


Cos' è il DMA

- DMA consente accesso diretto alla memoria da parte del disco con intervento minimo della CPU
 - I dati sono trasferiti direttamente dal dispositivo alla memoria e viceversa
 - Nel frattempo il processore è libero
 - C'è un chip I/O che gestisce il trasferimento al posto della CPU
 - La CPU viene notificata quando il trasferimento termina
 - Enorme miglioramento della performance per sistemi multitasking con molti accessi al disco



DIRECT MEMORY ACCESS

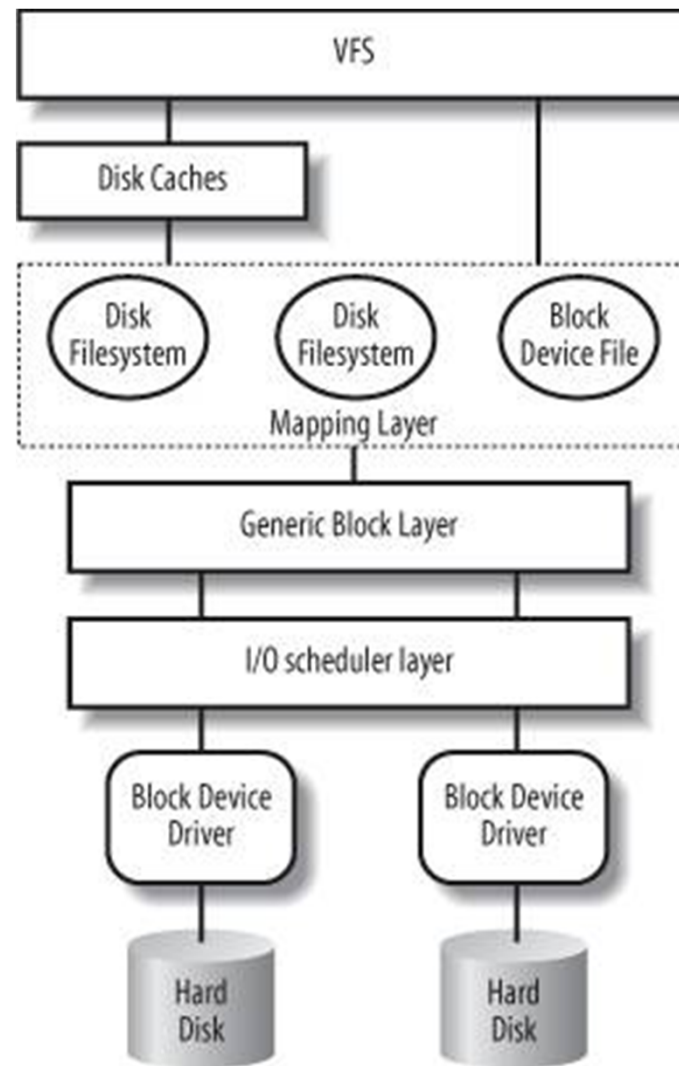


Come avvengono le notifiche alla CPU?

- Gli eventi hardware possono essere catturati e gestiti tramite gli interrupt
 - Ad esempio possono essere generati da dei processi
 - A causa di una divisione per zero
 - A causa di un tentativo di accesso a un area di memoria protetta
 - Ma anche, possono nascere da eventi esterni
 - Ad esempio un interrupt è generato se il mouse viene mosso
 - O se qualcuno preme un tasto sulla tastiera
- La scomoda alternativa è il *polling* continuo
 - Il processore controlla continuamente tutti i dispositivi



Block device driver & scheduler



Cos'è un file system

- ▶ Numerazione lineare: un mapping da CHS (3 coordinate) a un sistema a 1 coordinata.
- ▶ Partizione: un blocco di settori consecutivi nel sistema di numerazione lineare
 - Ogni partizione viene di solito 'formattata' con un certo file system a scelta: FAT16, FAT32, NTFS, EXT3, ecc.
- ▶ I file system consentono di gestire lo spazio su disco organizzando logicamente i dati
- ▶ Files
 - Un gruppo di *cluster* a cui è associato un nome e che contiene dati tra loro correlati (esempio un immagine)
- ▶ Directory
 - Un 'raccolgitore' che raggruppa più file e possibili sottodirectory in un elenco comune



Gerarchia dei dati

- ▶ Al più basso livello di astrazione, ci sono dei semplici bit, memorizzati sulla superficie del disco.
- ▶ Questi sono raggruppati in byte, quindi in settori di 512 byte
- ▶ Ogni file system raggruppa i settori in cluster (cluster da $8k = 16$ settori)
 - Un file da 10 byte occupa ALMENO un cluster!!!
- ▶ Ogni file ha un piano di numerazione interno (a partire dal byte 0) e una organizzazione dipendente dal suo formato.
- ▶ Non è detto che un file sia FISICAMENTE memorizzato su cluster consecutivi, nonostante il suo piano di numerazione interno lo faccia credere



Operazioni sui metadati

- Open
- Close
- Create
- Destroy
- Copy
- Rename
- List (fstat)



Operazioni sui contenuti

- ▶ Il contenuto di un file può essere manipolato con operazioni come:
 - Open/Close/Locking
 - Read
 - Write
 - Update
 - Insert
 - Delete



I file system

- Sono composti da codice e speciali strutture dati
 - Le **strutture dati** sono contenute in speciali settori della partizione, riservate a questi scopi. Ad esempio ci può essere una tabella dei cluster liberi.
 - Il **codice** è parte del sistema operativo e si occupa di gestire tali strutture dati, e di fornire ai programmi utente tutte le funzionalità per gestire file e cartelle
- Il file system è responsabile della gestione dei file, della loro integrità, delle modalità di accesso.



Caratteristiche dei file system

- Indipendenti dal dispositivo:
 - Che si tratti di una penna USB, di un dischetto o di un disco fisso, l'interfaccia esposta al programmatore è comune. I dettagli implementativi sono nascosti.
- Dovrebbero fornire funzionalità di backup e/o recovery per contrastare le possibili perdite di dati
- Possono fornire funzionalità di compressione e crittografia
- Forniscono la possibilità di associare a file e directory attributi quali diritti di accesso, data/ora di creazione/modifica, ecc.
- Forniscono meccanismi di file locking



Directory e link

- ▶ Directories:

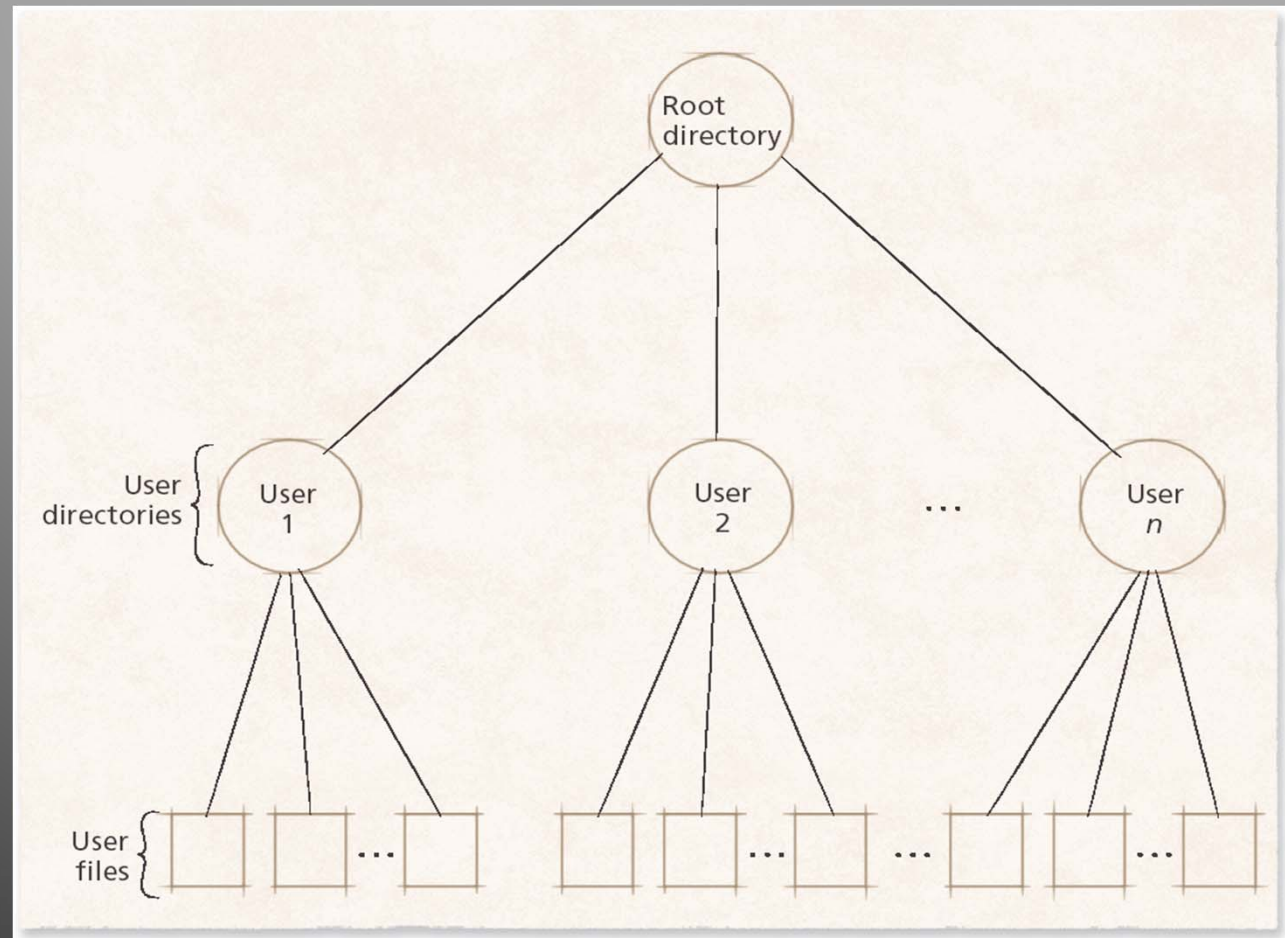
- Sono 'file' speciali che in realtà costituiscono dei raccoglitori che elencano altri file

- ▶ Link

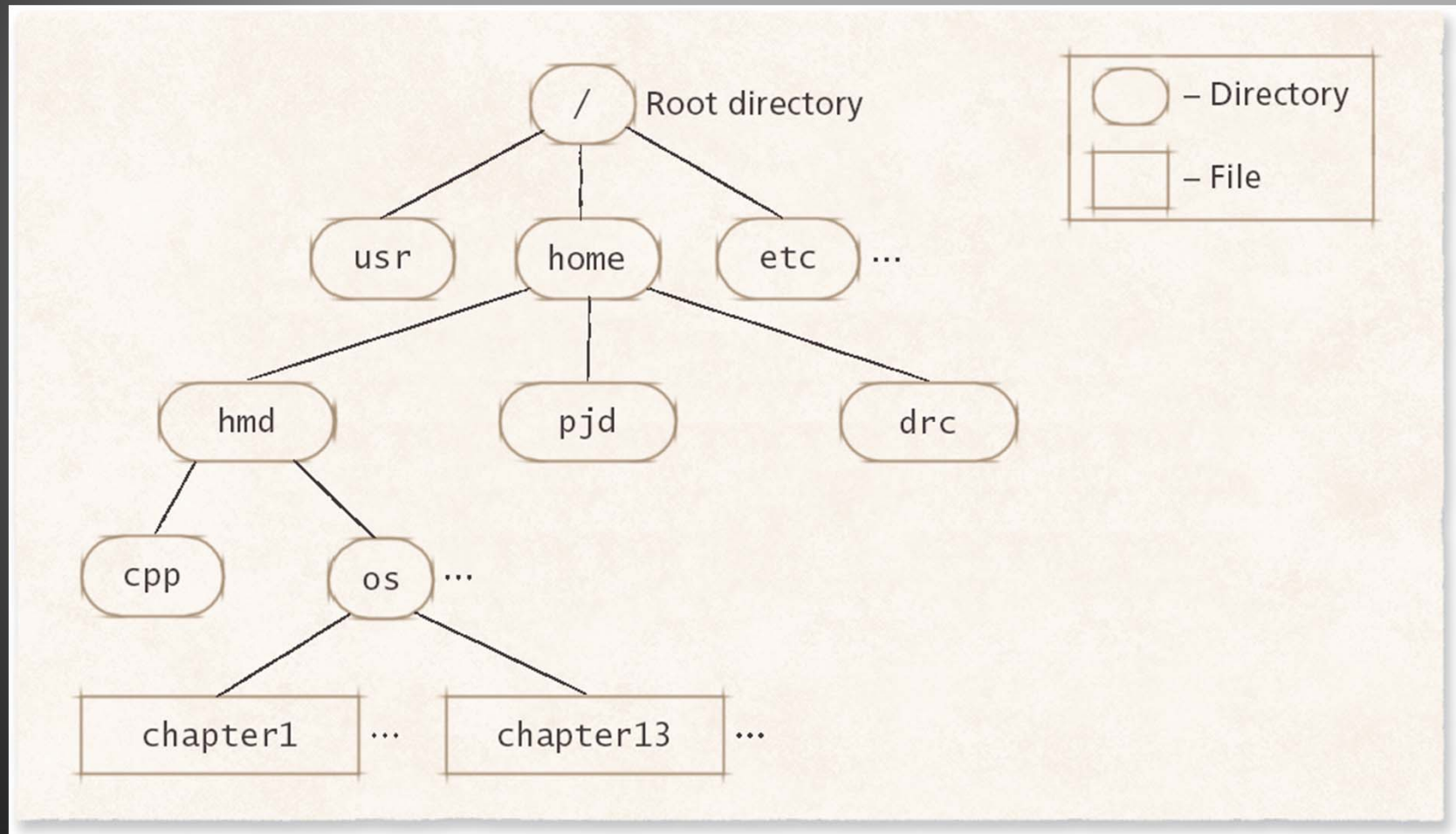
- Un voce che sta in una directory che fa riferimento a un file logicamente raggruppato in un'altra



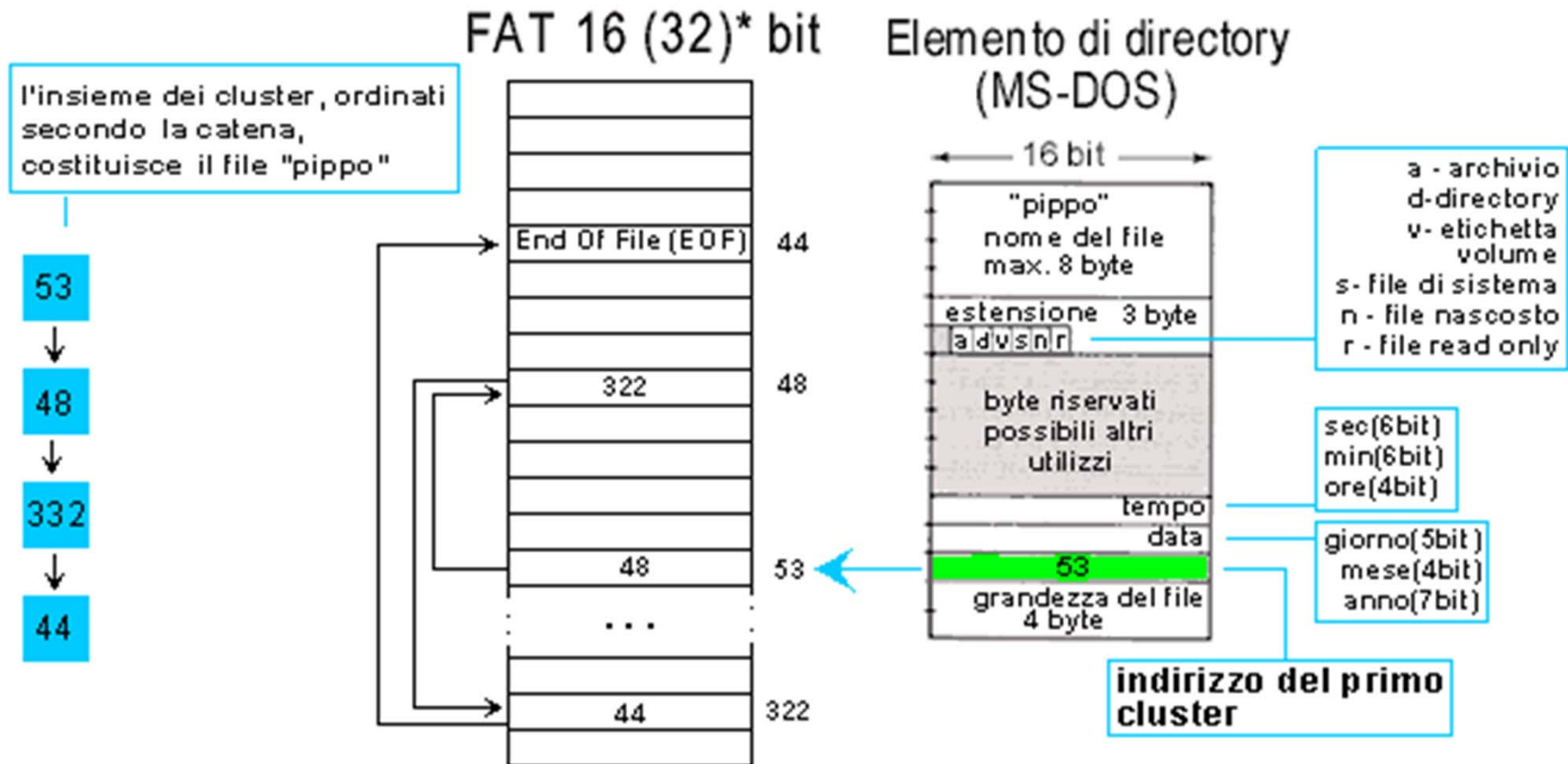
Directory gerarchiche

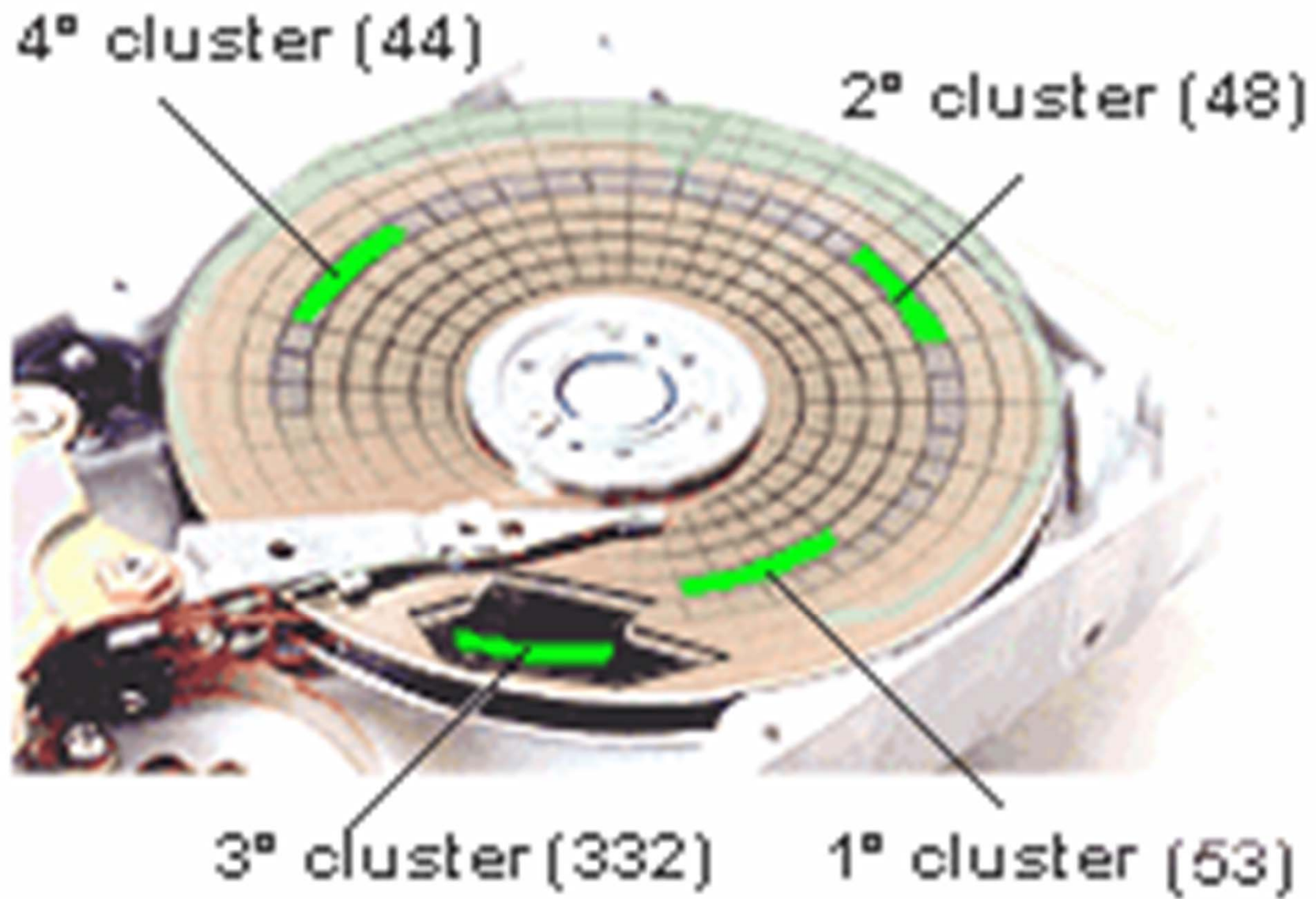


Directories



FAT32





File read()

- ▶ Unsigned long FAT[]
- ▶ File Directory[]
- ▶ Class File
 - { char* nome;
 - ...
Unsigned long startingBlock; }
byte* read(long length) { ... }



NTFS & Ext4

▶ NTFS

- Molto robusto
- Remapping automatico dei cluster deboli
- Compressione
- Crittografia
- Diritti di accesso sofisticati
- Transazionale (sempre consistente)

▶ Ext4

- journaled (transazionale)
- Ottima politica anti-frammentazione



File locking

▶ Windows

- Possibilità di aprire un file in modalità `SHARED_READ`, `SHARED_WRITE`, `SHARED_DELETE`
- Possibilità di mettere lock su porzioni di file

▶ Linux

- Assenza di lock esplicito
- I programmi devono *volontariamente* usare `lockf`
- Possibilità di montare file systems con lock obbligatorio



Il concetto di “mounting”

- Operazione di mount:
 - Combina più file system in un'unica gerarchia
 - Assegna una cartella, chiamata “mount point” nel file system nativo. Al mount point viene “agganciata” la radice del file system “montato”.
- Le “mount tables” memorizzano le informazioni sui punti di mount:
 - E' una tabella di associazioni tra “mount point” e partizioni/dispositivi
- La tabella è usata dal SO per capire che formato hanno i dati e che dispositivo deve essere montato per ciascun mount point



Esempi

Figure 13.5 Mounting a file system.

