

Course TÖL401G: Stýrikerfi / Operating Systems

12. Secondary Storage Structure

Mainly based on slides and figures
copyright Silberschatz, Galvin and Gagne

Chapter Objectives

- Describe the physical structures of various secondary storage devices and the effect of a device's structure on its uses.
- Explain the performance characteristics of mass-storage devices.
- Evaluate I/O scheduling algorithms.
- Discuss operating-system services provided for mass storage, *including RAID*.

Contents

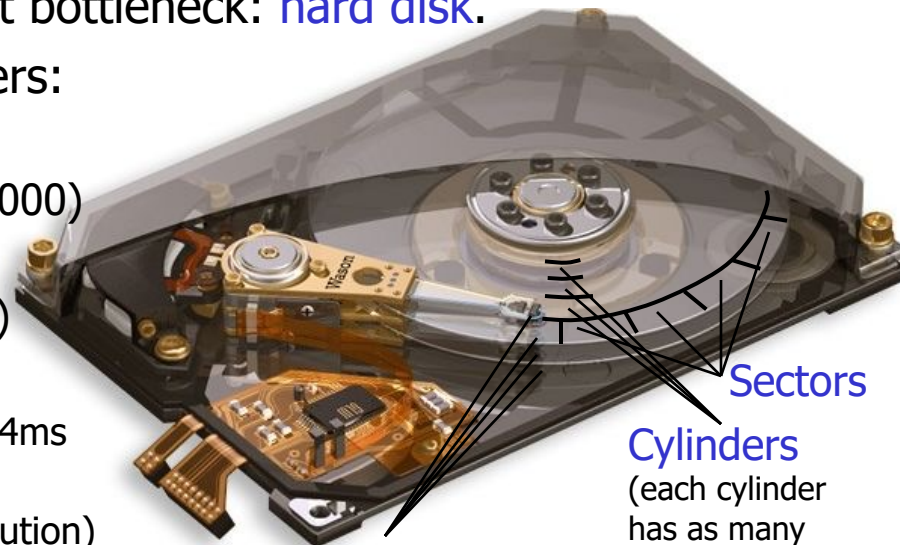
1. Introduction
2. Disk Structure
3. Storage Attachment
4. Disk Scheduling
5. Disk Management
6. Swap-Space Management
7. *RAID*
8. *Tertiary Storage*
9. Summary

12.1 Introduction: Hard Disks (HDs)

- Most important I/O device and biggest bottleneck: **hard disk**.

- Storage using rotating magnetic platters:

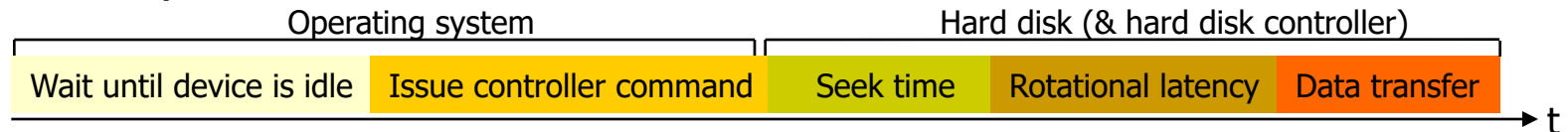
- **Heads (2 per platter)** (2–8)
- **Cylinders** (up to e.g. 200 000)
- **Sectors** (each 512 Byte or 4KB
e.g. $400 \times 0.5\text{KB} - 250 \times 4\text{KB}$ per track)
- Performance data:
 - Avg. head positioning seek time: 4 – 14ms
 - Rotational velocity:
 $4\,200 - 15\,000\text{ RPM}$ ($= 4 - 14\text{ms/revolution}$)
 - Data transfer bandwidth: $\approx 14 - 250\text{MByte/s}$



Read/write heads
attached to **disk arm** (moves
all heads together in one unit)

Sectors
Cylinders
(each cylinder
has as many
tracks as the
disk has heads.)

- Anatomy of access to random sector:



- Disk rotates and read/writes heads are positioned together:

- ⇒ First fill all sectors of the same track, then fill all tracks of the same cylinder, then move heads to next cylinder (moving head 1 cylinder = fast, multiple = slower).

Introduction: Solid-State Disks (SSDs) / Nonvolatile Memory (NVM) Devices

- (Flash-)RAM based storage.

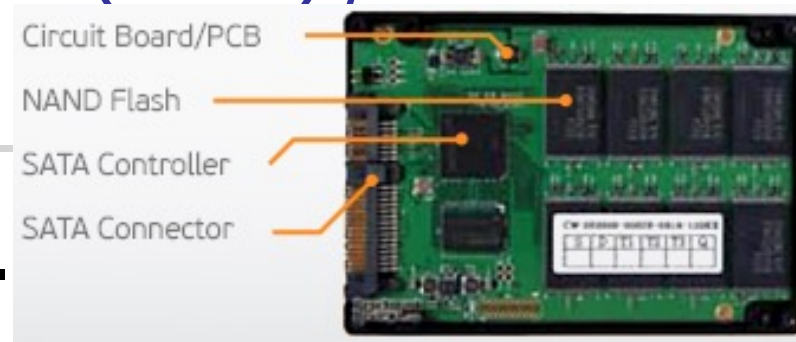
- No moving mechanical parts:

- No seek time, no rotational latency ⇒ **Faster** (≈100-1000 times).
- Lower energy consumption, no danger of head crash, no noise.
- **More expensive** ⇒ **Lower capacity**.

- Each block of a flash-based SSD can only be erased (and therefore written) a limited number of times before it fails.

- **Wear leveling**: imagine a block that stores the FAT or I-Nodes: changed/written very often: SSD varies the low-level physical flash memory cell used to store often written logical blocks.
- OS filesystem can tell the SSD via **TRIM command** which blocks are not used by the filesystem (SSD does not know filesystem): if SSD needs to erase flash cells it can use this information to erase flash cells in bigger chunks (flash block) which is faster.

Experience shows: wear leveling is so good, that 10-100 years of normal home/office use is not an issue – SSDs are far more reliable than HDs.

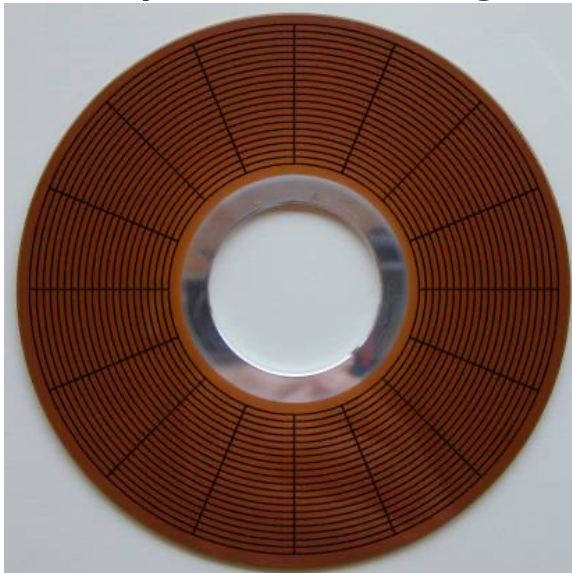


Source: <http://ocz.com/consumer/ssd-guide/ssd-vs-hdd>

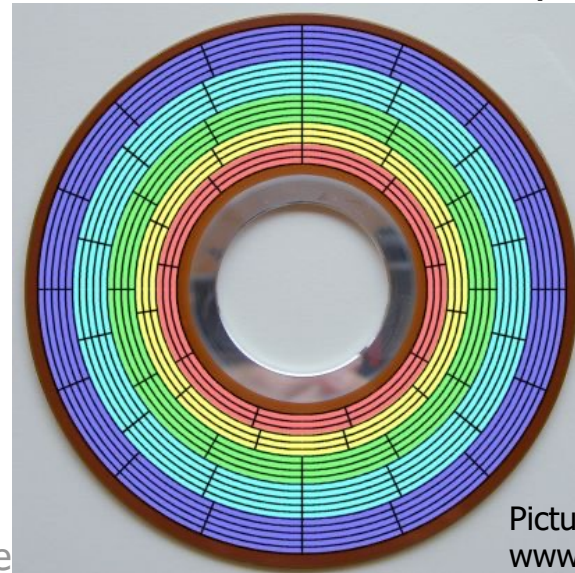
12.2 Hard Disk Structure: Velocity and Bit Density

- Platters rotate always with the same number of rounds per minute (in contrast to, e.g., CD/DVD), i.e. **constant angular velocity** (CAV).
- As the circumference of outer tracks is longer than of inner tracks, much more bits fit on outer tracks than on inner tracks (assuming that there is some maximum possible density of bits per inch).
- **Zone Bit Recording** (ZBR): store more sectors per track on outer tracks than on inner tracks. (Transfer rate higher on outer tracks: more data per rotation.)

CAV
without
ZBR



CAV
with
ZBR



Picture source:
www.storagereview.com

Hard Disk Structure: Sectors

Data of write() system call not exactly matching a full sector ("partial write"): OS has to read sector first, modify in memory, write sector back.

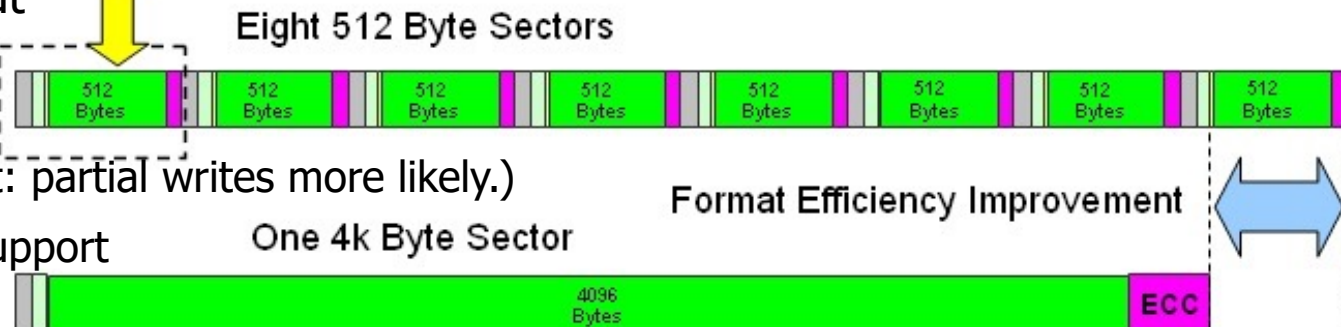
- Disk rotation speed has jitter \Rightarrow Changing just single bits during write access not possible as probably the wrong bit might be hit.
- Data always written (and read) in blocks (**sectors**).
 - Even if only 1 bit shall be changed/read, always full sector needs to be written/read.
 - Each sector starts with a **synchronisation** pattern (created by low-level formatting): Allows disk controller to detect exact start of a sector independent of jitter.
 - **Gap** between sectors: even if disk rotates slower during a later write (= physical space occupied by written sector increases), next sector will not get touched.
 - **Address mark**: sector&cylinder number: to know which sector&cylinder rotates along.

- Past decades:
512 Byte sectors.



Picture source: Wikipedia

- Since 2010:
"Advanced Format"
4 KB sector size.




Address mark=
Cylinder, head, sector number
ECC=Error-correcting code

- Reduced overhead. (But: partial writes more likely.)
- OS needs to support buffer size!

12.3 Storage Attachment

- Ways to connect secondary storage to a computer:
 - **Host-Attached Storage**: Local hard drive connected via short cable to a local I/O port (or SSD in M.2 card format plugged into M.2 slot).
 - E.g. ATA/IDE, NVM, SCSI, PCI Express, USB case internally using ATA.
 - **Network-Attached Storage (NAS)**: Storage provided via some file server that is accessed via TCP or UDP over an IP network.
 - E.g. Sun NFS or Microsoft SMB/CIFS Windows shares, i.e. **file-based access**.
 - Existing network is used (and shared with other applications) for attachment.
 - Storage of NAS server is either host-attached or attached via SAN.
 - **Storage-Area Network (SAN)**: A completely separate network infrastructure is used for connecting storage unit and computers.
 - E.g. FibreChannel (special type of cables and network adapters) or iSCSI (SCSI over IP network, allows using standard network hardware). **Block-based access**.
 - Dedicated network channels: not slowed down by ordinary network traffic.
 - Typically used only for attaching “big” storage to “big” servers (not for your PC).
 - Multiple storage units shared by multiple servers within the same SAN.

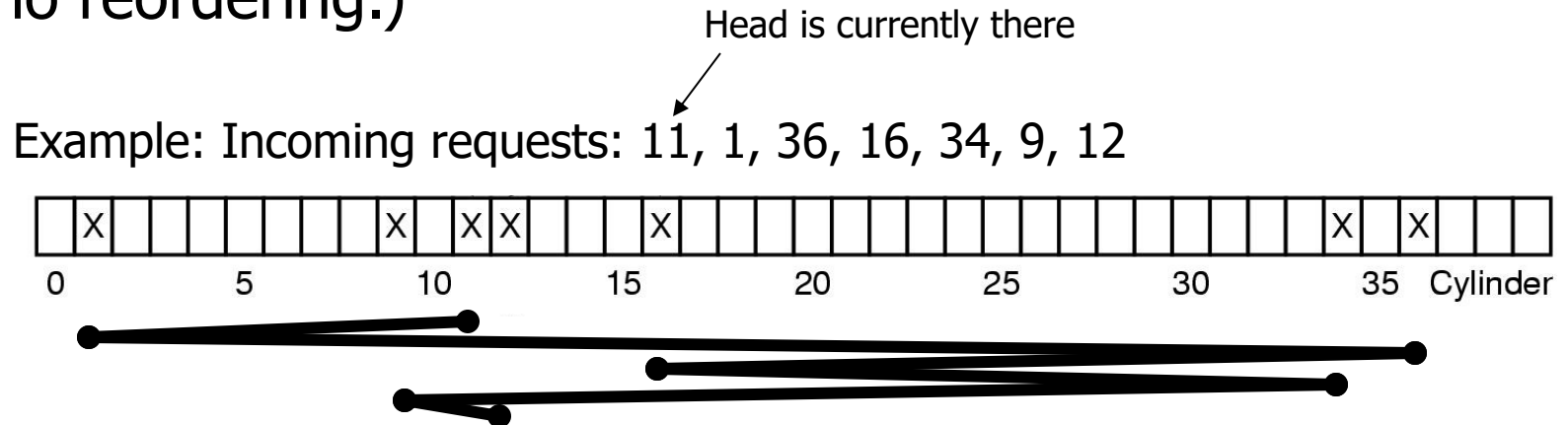
12.4 Disk Scheduling

- Goal: minimise “expensive” positioning of disk heads.
- Observation: In bottleneck situations, disk requests queue up at the operating systems disk device driver.
 - Idea: Disk driver rearranges order of queued-up disk access requests (=head positionings) to achieve a minimum number of head movements.
 - Reordering only possible if no dependencies between data transferred by disk access:
 - In fact, only accesses to the same sector may be a problem:
 - $A=\text{read}(\text{sektor42}), \text{write}(\text{sektor42}, B), C=\text{read}(\text{sektor42}): A \neq C, C=B$
 Reorder writing of B and reading into C
 - $A=\text{read}(\text{sektor42}), C=\text{read}(\text{sektor42}), \text{write}(\text{sektor42}, B): A=C \neq B$
- ⇒ Maintain order of accesses to the same sector.
- In practise, accesses to the same sector are already cached at the OS level, hence reordering on the lower disk level is no problem!

Disk Head Scheduling Strategy

First Come, First Served (FCFS)

- “Serve requests in the order of their arrival.” (I.e. FIFO, no reordering.)



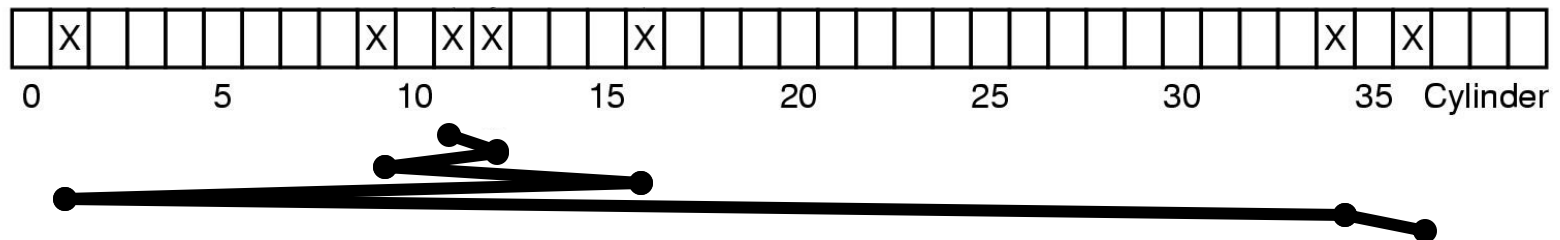
“Travel” distance of head: $10+35+20+18+25+3=111$ cylinders

- Assessment:
 - No minimisation of head “travel” distance.
 - + Fair (& easy to implement).

Shortest Seek First (SSF) / Shortest Seek Time First (SSTF)

- “Select from the queued-up request the one that is closest to the current disk head position, i.e. the request with the minimum seek time.” (→SJF CPU scheduling)

Example: Incoming requests: 11, 1, 36, 16, 34, 9, 12



“Travel” distance of head: $1+3+7+15+33+2=61$ cylinders

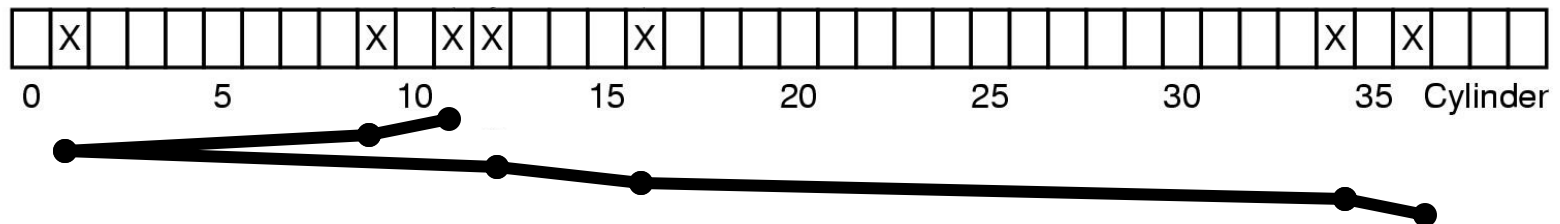
- Assessment:
 - + Less head “travel” distance than FCFS (but in general not optimal).
 - Unfair: starvation of requests possible (just like SJF) if new requests come in that are close to current disk head position.

Elevator Scheduling / Scan / Look

- “Move disk head back and forth between the smallest and largest cylinder value of the queued-up requests. Service all requests as the corresponding cylinder is reached.”

Head is currently at cylinder 11 and coming from larger cylinder numbers

Example: Incoming requests: 11, 1, 36, 16, 34, 9, 12



“Travel” distance of head: $2+8+11+4+18+2=45$ cylinders

- Note: Sometimes, Scan and Look are considered as different strategies: Look behaves as above; Scan constantly travels between the minimum and the maximum cylinder number (even if no requests exist).

- Assessment:

- + Less head “travel” distance than FCFS (but in general not optimal).
- + Fair (If incoming requests for the current cylinder are not processed immediately, but on the way back.)

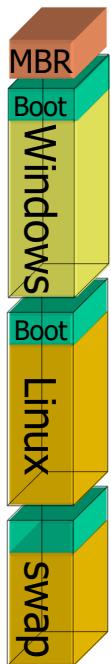
12.5 Disk Management:

Low-level Formatting & Bad Blocks

- Before a hard disk can store data, it needs to be **low-level formatted**:
 - **Synchronisation pattern**, **gap** (→12.2), **address mark** (so that the read/write head knows which sector is currently rotating along and also (when a head seek is made) to find out whether right cylinder has been hit or whether head repositioning is needed to find the right cylinder) are created.
 - If (due to defects in the magnetic coating) unusable sectors are found, these are marked as bad.
 - On each track, some **spare sectors** are reserved as replacement for bad sectors.
 - Bad sectors mapped to spare sectors (nowadays automatically by the hard disk).
 - Sectors becoming bad during usage are revealed by error correcting code (ECC).
 - If ECC cannot correct error anymore (**hard error**), sector is marked bad and **mapped to spare sector**.
 - As long as error can be corrected by ECC (**soft error**), sector is typically not mapped to spare sector.
 - In the past, OS needed to provide a low-level formatting system program.
 - Nowadays, each drive needs a different low-level formatting, hence the manufacturer ships an already low-level formatted hard disk.

Disk Management: Logical Formatting

- Disks are divided into **partitions**/volumes:
 - One file system per partition/volume.
 - ⇒ Multiple, different file systems possible per disk.
 - Creation of initial data structures for a file system on a partition/volume is called **logical formatting**.
 - Partitioning based on logical aspects (eases also backup), e.g.
 - File system 1 with operating system,
 - File system 2 for installing applications,
 - File system 3 with home directories containing user files,
 - File system 4 with for virtual memory ("swap partition").
 - Partitioning to install multiple operating systems at the same time ("Dual Boot"), e.g. on BIOS systems:
 - 1 Master Boot Record per hard disk (containing boot selector code),
 - For each file system: 1 boot sector (containing bootloader for OS).
 - On UEFI: dedicated FAT partition instead of Master Boot Record.



12.6 Swap-Space Management

- Two possible hard disk swap-/paging-space management approaches:
 - Only store those pages on disk that are currently not in physical memory.
 - Reduces needed size of swap space.
 - When a page in a frame is replaced it must be written to swap space even if it has not been modified. (Because as it was in physical memory, it was not stored in swap space and would be lost once it gets replaced in physical memory.)
 - Leave a copy of a page on disk even if it is currently in physical memory.
 - When an unmodified page in a frame is replaced, no need to write it to swap space. ⇒ Faster
 - Bigger swap space needed.
 - Linux does not page out instructions of a process, but simply reloads them from the executable.
- Either a “raw” partition on it’s own can be used as swap space.
 - Fast. Changing swap space size difficult (resizing a partition is difficult).
- or a huge file on an ordinary file system partition is used.
 - Ordinary file access is slow. Creating bigger swap file easy (as long as enough space is available in the file system).

12.9 Summary

- Major storage device: SSDs and still hard disks: predicted that hard disks will vanish in PCs, only to be found in huge storage centers.
 - Needed by file systems and virtual memory system.
 - Hard disk is random access, access speed influenced by mechanic properties.
 - Disk scheduling to speed up access of hard disk read/write heads.
 - (However, disk head scheduling is nowadays not reasonable at OS level: Disk drives may use a different layout than reported to the OS: cylinder boundaries may be different (e.g. due to ZBR), spare sectors that are used as replacement for bad sectors may be located at different locations.
⇒ Disk head scheduling done at disk level (high-end hard disks only).
 - Low-level formatting to create, e.g., synchronisation patterns.
 - SSDs have no mechanic parts (no need for head scheduling, low-level formatting), but need internally wear leveling to deal with limited number of erase/write accesses.
- Different ways of attaching storage to computers (Host, NAS, SAN).
- High-level formatting to create partitions containing file systems.
- RAID to speed up access (striping) and/or increase reliability (mirroring).
- Tertiary storage almost obsolete (now rather: back up SSD to hard disk).