Magnetic storage media

In Minix 3, the `at_wini` driver is used for hard disks, CDROMs and floppy drives.

A hard disk drive (HDD, sometimes just called "a hard disk" or "HD") is a magnetic storage medium. Its name derives from the fact that a HD consists of one or more circular platters (or disks) made of a rigid material covered in a magnetizable material.

This is in contrast to "floppy disks", which were a popular storage medium before hard disks became dominant. A floppy disk consists of a soft magnetizable disk enclosed in a plastic housing. Older floppy formats were larger and had non-rigid housings.

Most images removed from these slides because of copyright concerns.
Disk geometry

A hard disk consists of a number of **platters**. The platters stacked vertically on the same central axis, and they are rotated together by a electric motor in the hard disk.

Each platter is divided into concentric rings called **tracks**. Each track is divided into pieces called **sectors**.

Instead of viewing the disk as a set of platters with tracks, device drivers typically view a HD as a set of cylinders. Each **cylinder** is composed of all the tracks that are directly above or below each other on different platters.

A HD as a set of **heads** that skim across the surface of the platters. The heads read or write bits to the surface by sensing magnetic fields (read) or remagnetizing the material with a small electric pulse (write)
The International Business Machines Corporation (IBM) was a pioneer in the field of hard disk technology. One of IBM's hard disk development projects in the 1970's was codenamed "Winchester". Most hard drives in modern personal computers are based on ideas first developed by this project, just as the ubiquitous x86_64 CPU architecture can trace its roots to the IBM 80486 (nicknamed "the 486"). Consequently, some drivers, such as Minix' at_wini are still named after it.

The heads must come very close (typically 3 nanometers) to platters to read and write information, without actually touching the platter! "Winchester heads" skim accross the surface by riding on the air current formed by the spinning disk platter.
The geometry is a lie

In the past it was necessary for the disk driver software to be aware of all the nitty-gritty details of disk geometry (the way data is physically layed out on the magnetic platters).

Nowadays, the physical geometry is abstracted away from the device driver and even the device controller: modern hard disks support the Integrated Drive Electronics (IDE) standard. The electronics+firmware on the disk itself perform many complex tasks and mediate between the device controller and the physical disk motors, actuators and sensors.

This allows modern drives to support Logical Block Addressing (LBA). With LBA, the device driver does not have to refer to specific sectors in specific cylinders. Instead, it can simply ask the hard disk to read or write to a block identified by an index starting from zero, and the drive automatically calculates on which platter/track/sector the block resides.

This is a good example of separation of concerns: only the disk hardware/firmware is concerned with geometry, which makes disk driver and the controller interface simpler, and makes a single driver compatible with disks of any internal geometry as long as they support LBA.
The geometry is still used

In reality, disks typically use more advanced schemas such as block zoning. A modern hard disk has sectors of equal length, but fewer sectors near the center of the platters, and more near the outer rim.

Because awareness of geometry played such an important role in disk drivers in the past, and because newer standards and interfaces were based on older ones, most drivers still use geometry data structures and request this information from the disk.

In 3.6 we saw that the RAM disk's geometry is entirely fake, but it is still included. LBA-capable hard disks also report 'fake' geometry to the controller, and allows the controller (and hence the device driver) to access blocks via this fake geometry if it needs to (eg. older operating systems). Because of the way IBM set up the original geometry data structures, non-LBA access limits the usable disk space (8GB limit assuming 512byte sector size)
It must be possible to somehow refer to an address (verb) every usable location (noun) on a storage device.

Addresses are integer numbers (or compounds of integer numbers as in the case of cylinder/cluster/sector addressing). Think of pointers, for example.

If you have a storage device of 1000 sectors, how many bits wide must a variable be to hold an address that refers to a specific sector?

A 1-bit unsigned integer can hold the numbers 0 and 1. A 2-bit uint can hold values 0 to 3 (binary 00 to 11). A 3-bit uint can hold values 0 to 7 (000 to 111).

See the pattern? Adding one bit makes the variable capable of holding a range twice as large as it would have without the additional bit.

Number of distinct values holdable by variable = 2 to the power of [number of bits]

9-bit uint can hold 0-511, but a 10-bit uint can hold 0-1023, which is big enough to address a range of 1000 sectors.
The units are a lie (sort of)

A 10-bit number can be used to address (v) an address-space (n) of 1000 sectors. But what about the other 24 values? They are essentially "wasted". System designers therefore traditionally made storage exactly big enough to be addressable by an integer number of bits, and therefore defined units in terms of powers of two:

1 Kilo-Byte = 1KB = $2^{10}$ bytes = 1024 bytes
1MB = 1024KB; 1GB = 1024MB; 1TB = 1024GB; 1PB = 1024TB; etc.

However! In other engineering disciplines, the kilo-, mega-, giga-, etc. prefixes are used to indicate a multiplier of 1000 and not 1024 (ie. other disciplines use base-10 units as opposed to base-2 units)

In computer science, power-two units are generally more useful.

Hard disk manufacturers saw an opportunity to make their disks look bigger to buyers, by labelling their disks with base-10 units

10GB ($10^9$) = 1000000000 bytes, which is 7.3% smaller than 10GiB ($2^{30}$) = 1073741824 bytes thus $10\text{GB} = 9.31\text{GiB}$
To read or write a sector from a platter, the hard disk must:

1. determine the head to use and the cylinder and sector number
   - This is done by the IDE microprocessor on the drive. Very fast.
2. position the head stack onto the correct cylinder by moving the "arm"
   - IDE electronics power the arm's motor. This is the slowest step by far (can take several milliseconds)
3. rotate the platters until the desired sector comes under the chosen head
   - A HD generally keeps spinning its platters at a constant speed as long as the drive is switched on, so this does not take long (only a matter of waiting for the correct sector to come under the head).
4. read or change the magnetic pattern of the bits in that sector
   - By pulsing an electric current through the head's electromagnetic coil. This is also relatively fast.

It is worthwhile to optimize HD arm movement to improve speed.
Remaining topics

These are covered by the book and the 2013 slides.

- Error checking: Cyclic Redundancy Check (#14)
- Disk arm scheduling: FCFS, shortest seek first and elevator algorithm (#15)
- Error handling (#23)
- Hard disk caching (notably track-at-a-time caching) (#24)
- RAID (#9)
- Hard disk driver and boot sequence (#26)
- Floppy disks in Minix (book)
- Minix driver internal details