Where we were

- Transfer data to/from devices
  - I/O instructions: `inb/inw/outb/outw`
  - Memory mapped I/O
  - DMA

- Coordination w/devices

- Interrupts
  - Don't waste cycles polling
  - High interrupt rates lead to no forward progress

- Polling
  - Might waste cycles
  - OS decides when to check
  "→ Forward progress"

- In practice
Adaptive

Interrupt Mode

Poll mode

No I/O for k-polls

Device Drivers

Interfaces

Vary

GPU A

GPU B

SSD

Kernel

Provides Common Interface

Knows GPU A's Interface

GPU A Driver

Knows GPU B's Interface

GPU B Driver

Synchronous vs Asynchronous I/O

FROM USERSPACE

So far: read/write... block the calling thread

A condition: Thread has nothing to
Assumption: Instead of waiting do while waiting

But, what if thread could do other things (e.g., switch to run different usermode thread)

Most operating systems offer non-blocking APIs Not standard

Do not need to know

\[
\text{\{ - Linux: epoll, io-wiring
- OS X: kqueue
- Windows: Completion ports \}}
\]

Similarity: Poll for completion

\rightarrow Mechanism to wait/block if necessary

**Disks**: Spinning Disks

Why?
- Still widely used
- Dictate the design of most/many file systems
Interface: Linear array of sectors (generally 512 bits)

Read/Write 4K starting from

How: Disk Controller translates sector

(Implemented by disk hardware)

Read (25)

Head

Comparison

0.10
Total Transfer Time = Rotational Delay + Seek Delay + Transfer Time

Disk Rotation
E.g. 7200 RPM
10000 RPM

Avg. Rotation time from RPM
7200 RPM = 120 RPS

⇒ 1 Rotation every
$\sqrt{120} \approx 8.3$ ms per rotation

⇒ Avg. Rotation time = 4.05 ms
10000 RPM $\Rightarrow$ 167 RPS
$\Rightarrow$ $\sim$ 6 ms per rotation
$\Rightarrow$ Avg Rotation time $= 3$ ms

Let us use this:

Disk:
- Rotation: 12,000 RPM
- Avg. Seek Time: 12 ms
- Transfer rate: 128 MB/s
- Assume 512 Byte reads: 1 512 B sectors

Q) Throughput (bytes/second) to read 500 sectors spread randomly across disk, served in FIFO order?

Read 27, 12, 100, 17, ...

Might need to seek for each sector

$\text{Throughput} = \frac{\text{Data Read}}{\text{Time taken}}$

Data read $= 500 \times 512$ B
Time taken for 1 read (on avg)

Seek time = 12 ms

Rotation time = 2.5 ms

12000 RPM = 200 RPS

⇒ 1 rotation every \( \frac{1}{200} \) s = 5 ms

⇒ Avg Rot time = 2.5 ms

\[
\text{TRANSFER} = \frac{512 \text{B}}{128 \text{MB/s}} = \frac{2^9}{2^7 \cdot 2^{10} \cdot 2^0}
\]

\[
= \frac{1}{2^{18}} \approx 4 \mu s
\]

\[
\text{Time for 1} = 12 \text{ms} + 2.5 \text{ms} + 0.004 \text{ms}
\]

\[
\text{TRANSFER} \approx 14.5 \text{ms}
\]

\[
\text{TPUT} = \frac{500 \times 512 \text{B}}{500 \times 14.5 \text{ms}}
\]
\[ \text{Throughput for 500 sequential sectors?} \]

5, 6, 7, ... 505

What changes? - seek once to track, wait once for head to arrive at 5.

\[
\text{Seek time} = 12 \text{ms} \\
\text{Rotation time} = 2.5 \text{ms} \\
\text{Transfer time} = \frac{500 \cdot 2^9 \text{B}}{2 \cdot 2^{20} \text{B/s}} = \frac{500}{2^{18}} \text{s} = \frac{2^9}{2^{18}} \text{s} = \frac{1}{512} \text{s} \\
\approx 2 \text{ms}
\]

\[ \text{Throughput} = \frac{500 \cdot 2^9 \text{B}}{16.5 \text{ms}} = \frac{500 \cdot 10^3 \cdot 2^9 \text{B/s}}{16.5} \]
Observation: \textit{Sequential Access Is Much Faster}.

\textit{Constraint For A Lot Of File System Design.}