How do users interact with disks?

Storage?

reading/writing Files

```c
int fd = open ("myfile.txt", O_RDONLY);
```

- Call to the operating system
- File descriptor
- Index into a table in the OS

- File descriptor table
- Open file table
- Shared by all processes
- per-process basis

- File A
- File B
- File C

- Mode table
- Access
- Size
- File type
- To disk
Process -> application
  
  The running code + other things that go with it on OS

  File descriptor table -> per process
  Open file table -> one in OS -> dynamic info. per open file
  V-node table -> holds all info about the file, including physical location

File system specific -> static information per file

read (fd, buf, 100);
What actually happens?
1) Read asks the OS to do it (syscall)

2) Request a sector from the disk

3) Disk writes data to memory (DMA)

4) Copy data into buf

Direct memory access OS
How to send a message to disk?

- Pick an address for I/O
- Memory-mapped I/O
  - MMIO
- PIO - programmed I/O
  - `inb`, `outb`
  - `read`, `write`

```c
while (STATUS == Busy) {
  // Spin
}
```

- `spin loop`
- Called Polling the device
  - May need a timeout
  - Inefficient
  - CPU can't do anything else
  - Waste lots of energy
Solution?

Interrupt the cpu

Interrupts asynchronously

change control flow

int. handler:

movl (%esp), %eax
addl %ecx, %eax
mull %ecx
movl %eax, 9(%esp)

Interrupt

Interrupts only happen @ inst. boundaries
- precise interrupts