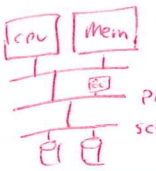
LECTURE

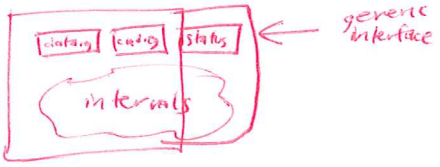
DEVICES + DISKS

Devices



device: h/w device driver: s/w (in OS)

↳ has interface



generic protocol:

```
while (status == busy)
    ; // poll device (wait for free)
write data → data reg
write cmd → cmd reg (device: go!)
while (status == busy)
    ; // poll (wait for done)
```

put caller to sleep

problems: (what is problem w/ polling?)

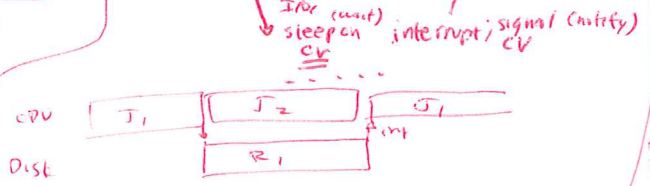
inefficient:
- → poll
- → data xfer

how to interact?
- → w/ device

→ interrupts:

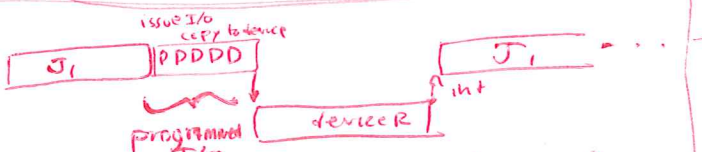
OS @ boot: program ICntrl (inform h/w of location of int handler)

then:



(1)

(2)



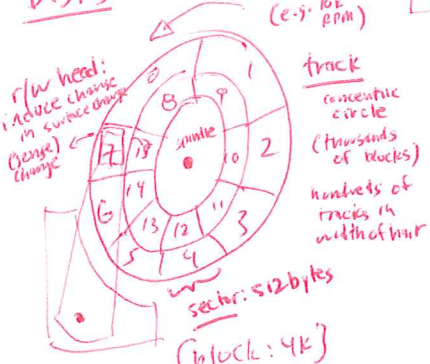
(CPU involved w/ each byte move)

alternate: DMA direct memory access (gives device this)

again: free CPU to do something else

principle: overlap!

Disks



may have > 1 platter

each has 2 surfaces

other tips:
- skew
- cache
- zones

to read/write:

- seek: move arm to correct track (speedup, coast, slowdown, settle)
- rotate: wait for correct block to come by
- transfer: do read/write of sector

$$T_{I/O} = \underbrace{T_{seek} + T_{rotate}}_{\text{bad: try to minimize}} + T_{transfer}$$

e.g. Sequential I/O (large block) vs. Random I/O (small blocks)

$R_{transfer}$: 100 MB/s

T_{seek}, T_{rot}: 10 ms

size: 10 MB vs. 10 KB

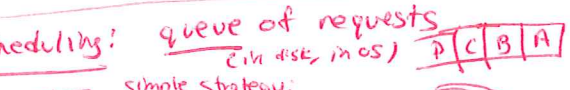
$T_{I/O} = 10ms + \frac{10\,MB}{100\,MB/s} = 110\,ms$

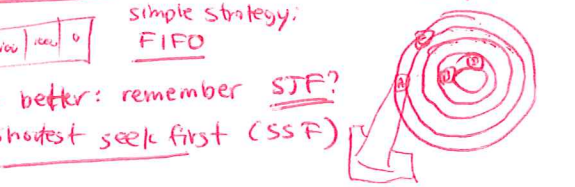
$T_{I/O} = 10ms + \frac{10\,KB}{100\,MB/s} \approx .2\,ms \approx 10\,ms$

$R_{I/O}: \frac{10\,MB}{110ms} \Rightarrow \approx 90.9\,MB/s$ $R_{I/O}: \approx 1\,MB/s$

(~100x difference!)

Disk scheduling: queue of requests (in disk, in OS)



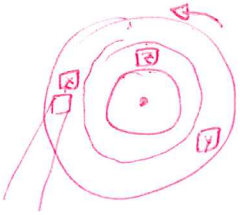


simple strategy: FIFO

can be better: remember SJF? shortest seek first (SSF)

problem 1: starvation ⇒ elevator (scan)

problem 2: not best possible: rotation!



X, Y, Z: order of ops?