Recall: Memory-Mapped Display Controller

- Memory-Mapped:
  - Hardware maps control registers and display memory into physical address space
    » Addresses set by hardware jumpers or programming at boot time
  - Simply writing to display memory (also called the “frame buffer”) changes image on screen
    » Addr: 0x8000F000—0x8000FFFF
  - Writing graphics description to command-queue area
    » Say enter a set of triangles that describe some scene
      » Addr: 0x8000F000—0x8000FFFF
  - Writing to the command register may cause onboard graphics hardware to do something
    » Say render the above scene
      » Addr: 0x0007F004
- Can protect with address translation
Transferring Data To/From Controller

- **Programmed I/O:**
  - Each byte transferred via processor in/out or load/store
  - Pro: Simple hardware, easy to program
  - Con: Consumes processor cycles proportional to data size

- **Direct Memory Access:**
  - Give controller access to memory bus
  - Ask it to transfer data blocks to/from memory directly

- Sample interaction with DMA controller (from OSC):
Goals for Today

• Finish discussion of device interfaces
• Discussion of performance
• Disks and SSDs
  - Hardware performance parameters
  - Queuing Theory

Note: Some slides and/or pictures in the following are adapted from slides ©2005 Silberschatz, Galvin, and Gagne. Many slides generated from my lecture notes by Kubiatowicz.
Basic Performance Concepts

- **Response Time or Latency**: Time to perform an operation (s)
- **Bandwidth or Throughput**: Rate at which operations are performed (op/s)
  - Files: mB/s, Networks: mb/s, Arithmetic: GFLOP/s
- **Start up or “Overhead”**: time to initiate an operation
- Most I/O operations are roughly linear
  - Latency \((n) = \text{Ovhd} + n/\text{Bandwidth}\)
Example (fast network)

- Consider a gpbs link (125 MB/s)
  - With a startup cost $S = 1$ ms
Example: at 10 ms startup (like Disk)
What determines peak BW for I/O?

- **Bus Speed**
  - PCI-X: 1064 MB/s = 133 MHz x 64 bit (per lane)
  - ULTRA WIDE SCSI: 40 MB/s
  - Serial Attached SCSI & Serial ATA & IEEE 1394 (firewire): 1.6 Gbps full duplex (200 MB/s)
  - USB 1.5 – 12 MB/s

- **Device Transfer Bandwidth**
  - Rotational speed of disk
  - Write / Read rate of NAND flash
  - Signaling rate of network link

- Whatever is the bottleneck in the path
Storage Devices

• Magnetic disks
  - Storage that rarely becomes corrupted
  - Large capacity at low cost
  - Block level random access (except for SMR - later!)
  - Slow performance for random access
  - Better performance for streaming access

• Flash memory
  - Storage that rarely becomes corrupted
  - Capacity at intermediate cost (50x disk ???)
  - Block level random access
  - Good performance for reads; worse for random writes
  - Erasure requirement in large blocks
  - Wear patterns
Are we in an inflection point?
Hard Disk Drives (HDDs)

Western Digital Drive
http://www.storagereview.com/guide/

IBM Personal Computer/AT (1986)
30 MB hard disk - $500
30-40ms seek time
0.7-1 MB/s (est.)

IBM/Hitachi Microdrive

Read/Write Head Side View
The Amazing Magnetic Disk

- Unit of Transfer: Sector
  - Ring of sectors form a track
  - Stack of tracks form a cylinder
  - Heads position on cylinders
- Disk Tracks ~ 1µm (micron) wide
  - Wavelength of light is ~ 0.5µm
  - Resolution of human eye: 50µm
  - 100K on a typical 2.5” disk
- Separated by unused guard regions
  - Reduces likelihood neighboring tracks are corrupted during writes
  - Track length varies across disk
  - Outside: More sectors per track, higher bandwidth
  - Disk is organized into regions of tracks with same # of sectors/track
  - Only outer half of radius is used
    » Most of the disk area in the outer regions of the disk
- New: Shingled Magnetic Recording (SMR)
  - Overlapping tracks ⇒ greater density, restrictions on writing
  - Seagate (8TB), Hitachi (10TB)
Magnetic Disk Characteristic

- **Cylinder**: all the tracks under the head at a given point on all surfaces
- **Read/write**: three-stage process:
  - Seek time: position the head/arm over the proper track (into proper cylinder)
  - Rotational latency: wait for the desired sector to rotate under the read/write head
  - Transfer time: transfer a block of bits (sector) under the read-write head
- **Disk Latency** = Queuing Time + Controller time + Seek Time + Rotation Time + Xfer Time

**Diagram**:

- Request
- Software Queue (Device Driver)
- Hardware Controller
- Media Time (Seek+Rot+Xfer)
- Result

- **Highest Bandwidth**:
  - Transfer large group of blocks sequentially from one track
## Typical Numbers for Magnetic Disk

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Info / Range</th>
</tr>
</thead>
</table>
| **Space/Density**           | Space: 8TB (Seagate), 10TB (Hitachi) in 3½ inch form factor! (Introduced in Fall of 2014)  
Areal Density: ≥ 1Terabit/square inch! (SMR, Helium, …) |
| **Average seek time**       | Typically 5-10 milliseconds.  
Depending on reference locality, actual cost may be 25-33% of this number.                                                             |
| **Average rotational latency** | Most laptop/desktop disks rotate at 3600-7200 RPM  
(16-8 ms/rotation). Server disks up to 15,000 RPM.  
Average latency is halfway around disk yielding corresponding times of 8-4 milliseconds |
| **Controller time**         | Depends on controller hardware                                                                                                                                             |
| **Transfer time**           | Typically 50 to 100 MB/s.  
Depends on:  
• Transfer size (usually a sector): 512B - 1KB per sector  
• Rotation speed: 3600 RPM to 15000 RPM  
• Recording density: bits per inch on a track  
• Diameter: ranges from 1 in to 5.25 in |
| **Cost**                    | Drops by a factor of two every 1.5 years (or even faster).  
$0.03-0.07/GB in 2013 |
Disk Performance Example

• Assumptions:
  - Ignoring queuing and controller times for now
  - Avg seek time of 5ms,
  - 7200RPM ⇒ Time for rotation: 60000(ms/M)/7200(rev/M) ~ 8ms
  - Transfer rate of 4MByte/s, sector size of 1 Kbyte ⇒
    1024 bytes/4×10^6 (bytes/s) = 256 × 10^-6 sec ≅ .26 ms

• Read sector from random place on disk:
  - Seek (5ms) + Rot. Delay (4ms) + Transfer (0.26ms)
  - Approx 10ms to fetch/put data: 100 KByte/sec

• Read sector from random place in same cylinder:
  - Rot. Delay (4ms) + Transfer (0.26ms)
  - Approx 5ms to fetch/put data: 200 KByte/sec

• Read next sector on same track:
  - Transfer (0.26ms): 4 MByte/sec

• Key to using disk effectively (especially for file systems) is to minimize seek and rotational delays
Intelligence in the controller

• Sectors contain sophisticated error correcting codes
  – Disk head magnet has a field wider than track
  – Hide corruptions due to neighboring track writes

• Sector sparing
  – Remap bad sectors transparently to spare sectors on the same surface

• Slip sparing
  – Remap all sectors (when there is a bad sector) to preserve sequential behavior

• Track skewing
  – Sector numbers offset from one track to the next, to allow for disk head movement for sequential ops

• ...
Solid State Disks (SSDs)

- 1995 – Replace rotating magnetic media with non-volatile memory (battery backed DRAM)
- 2009 – Use NAND Multi-Level Cell (2 or 3-bit/cell) flash memory
  - Sector (4 KB page) addressable, but stores 4-64 “pages” per memory block
  - Trapped electrons distinguish between 1 and 0
- No moving parts (no rotate/seek motors)
  - Eliminates seek and rotational delay (0.1-0.2ms access time)
  - Very low power and lightweight
  - Limited “write cycles”
- Rapid advance in capacity and cost ever since
SSD Architecture - Reads

- **Read 4 KB Page**: ~25 usec
- **No seek or rotational latency**
- **Transfer time**: transfer a 4KB page
  - SATA: 300-600MB/s => ~4 x10^3 b / 400 x 10^6 bps => 10 us

- **Latency = Queuing Time + Controller time + Xfer Time**
- **Highest Bandwidth**: Sequential OR Random reads
SSD Architecture - Writes (I)

• Writing data is complex! (~200µs – 1.7ms)
  - Can only write empty pages in a block
  - Erasing a block takes ~1.5ms
  - Controller maintains pool of empty blocks by coalescing used pages (read, erase, write), also reserves some % of capacity
• Rule of thumb: writes 10x reads, erasure 10x writes

Amusing calculation: is a full Kindle heavier than an empty one?

- Actually, “Yes”, but not by much
- Flash works by trapping electrons:
  - So, erased state lower energy than written state
- Assuming that:
  - Kindle has 4GB flash
  - $\frac{1}{2}$ of all bits in full Kindle are in high-energy state
  - High-energy state about $10^{-15}$ joules higher
  - Then: Full Kindle is 1 attogram ($10^{-18}$ gram) heavier (Using $E = mc^2$)
- Of course, this is less than most sensitive scale (which can measure $10^{-9}$ grams)
- Of course, this weight difference overwhelmed by battery discharge, weight from getting warm, ....
### Storage Performance & Price (Jan 13)

<table>
<thead>
<tr>
<th></th>
<th>Bandwidth (Sequential R/W)</th>
<th>Cost/GB</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDD²</td>
<td>50-100 MB/s</td>
<td>$0.03-0.07/GB</td>
<td>2-4 TB</td>
</tr>
<tr>
<td>SSD¹,²</td>
<td>200-550 MB/s (SATA)</td>
<td>$0.87-1.13/GB</td>
<td>200GB-1TB</td>
</tr>
<tr>
<td></td>
<td>6 GB/s (read PCI)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.4 GB/s (write PCI)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRAM²</td>
<td>10-16 GB/s</td>
<td>$4-14*/GB</td>
<td>64GB-256GB</td>
</tr>
</tbody>
</table>

*SK Hynix 9/4/13 fire


**BW:** SSD up to x10 than HDD, DRAM > x10 than SSD

**Price:** HDD x20 less than SSD, SSD x5 less than DRAM
SSD Summary

• Pros (vs. hard disk drives):
  - Low latency, high throughput (eliminate seek/rotational delay)
  - No moving parts:
    » Very light weight, low power, silent, very shock insensitive
  - Read at memory speeds (limited by controller and I/O bus)

• Cons
  - Small storage (0.1-0.5x disk), expensive (20x disk ???)
    » Hybrid alternative: combine small SSD with large HDD
  - Asymmetric block write performance: read pg/erase/write pg
    » Controller garbage collection (GC) algorithms have major effect on performance
  - Limited drive lifetime
    » 1-10K writes/page for MLC NAND
    » Avg failure rate is 6 years, life expectancy is 9-11 years

• These are changing rapidly
What goes into startup cost for I/O?

- Syscall overhead
- Operating system processing
- Controller Overhead
- Device Startup
  - Mechanical latency for a disk
  - Media Access + Speed of light + Routing for network
- Queuing (next topic)
I/O Performance

Response Time = Queue + I/O device service time

- Performance of I/O subsystem
  - Metrics: Response Time, Throughput
  - Effective BW per op = transfer size / response time
    » EffBW(n) = n / (S + n/B) = B / (1 + SB/n )
  - Contributing factors to latency:
    » Software paths (can be loosely modeled by a queue)
    » Hardware controller
    » I/O device service time

- Queuing behavior:
  - Can lead to big increases of latency as utilization increases
  - Solutions?
A Simple Deterministic World

• Assume requests arrive at regular intervals, take a fixed time to process, with plenty of time between ...
• Service rate ($\mu = 1/T_S$) - operations per sec
• Arrival rate: ($\lambda = 1/T_A$) - requests per second
• Utilization: $U = \lambda/\mu$, where $\lambda < \mu$
A Ideal Linear World

- What does the queue wait time look like?
  - Grows unbounded at a rate $\sim (T_A/T_S)$ till request rate subsides
A Bursty World

- Requests arrive in a burst, must queue up till served
- Same average arrival time, but almost all of the requests experience large queue delays
- Even though average utilization is low
So how do we model the burstiness of arrival?

- Elegant mathematical framework if you start with exponential distribution
  - Probability density function of a continuous random variable with a mean of $1/\lambda$
    - $f(x) = \lambda e^{-\lambda x}$
  - “Memoryless”
    - Likelihood of an event occurring is independent of how long we’ve been waiting
    - Lots of short arrival intervals (i.e., high instantaneous rate)
    - Few long gaps (i.e., low instantaneous rate)
Background: General Use of random distributions

- Server spends variable time with customers
  - Mean (Average) \( m_1 = \sum p(T) \times T \)
  - Variance \( \sigma^2 = \sum p(T) \times (T - m_1)^2 = \sum p(T) \times T^2 - m_1^2 \)
  - Squared coefficient of variance: \( C = \frac{\sigma^2}{m_1^2} \)
    Aggregate description of the distribution.

- Important values of \( C \):
  - No variance or deterministic \( \Rightarrow C = 0 \)
  - "memoryless" or exponential \( \Rightarrow C = 1 \)
    - Past tells nothing about future
    - Many complex systems (or aggregates) well described as memoryless
  - Disk response times \( C \approx 1.5 \) (majority seeks < avg)
What about queuing time??
- Let’s apply some queuing theory
- Queuing Theory applies to long term, steady state behavior ⇒ Arrival rate = Departure rate

Arrivals characterized by some probabilistic distribution
Departures characterized by some probabilistic distribution
Little’s Law

- In any **stable** system
  - Average arrival rate = Average departure rate
- the average number of tasks in the system (N) is equal to the throughput (B) times the response time (L)
- \( N \text{ (ops)} = B \text{ (ops/s)} \times L \text{ (s)} \)
- Regardless of structure, bursts of requests, variation in service
  - instantaneous variations, but it washes out in the average
  - Overall requests match departures
A Little Queuing Theory: Some Results

• Assumptions:
  - System in equilibrium; No limit to the queue
  - Time between successive arrivals is random and memoryless

- \( \lambda \): mean number of arriving customers/second
- \( T_{ser} \): mean time to service a customer ("m1")
- \( C \): squared coefficient of variance = \( \sigma^2/m1^2 \)
- \( \mu \): service rate = \( 1/T_{ser} \)
- \( u \): server utilization (0 ≤ u ≤ 1): \( u = \lambda/\mu = \lambda \times T_{ser} \)

• Parameters we wish to compute:
  - \( T_q \): Time spent in queue
  - \( L_q \): Length of queue = \( \lambda \times T_q \) (by Little’s law)

• Results:
  - Memoryless service distribution (\( C = 1 \)):
    » Called M/M/1 queue: \( T_q = T_{ser} \times u/(1 - u) \)
  - General service distribution (no restrictions), 1 server:
    » Called M/G/1 queue: \( T_q = T_{ser} \times \frac{1}{2}(1+C) \times u/(1 - u) \)
A Little Queuing Theory: An Example

- **Example Usage Statistics:**
  - User requests 10 x 8KB disk I/Os per second
  - Requests & service exponentially distributed (C=1.0)
  - Avg. service = 20 ms (From controller+seek+rot+trans)

- **Questions:**
  - How utilized is the disk?
    » Ans: server utilization, \( u = \lambda T_{\text{ser}} \)
  - What is the average time spent in the queue?
    » Ans: \( T_q \)
  - What is the number of requests in the queue?
    » Ans: \( L_q \)
  - What is the avg response time for disk request?
    » Ans: \( T_{\text{sys}} = T_q + T_{\text{ser}} \)

- **Computation:**
  \[
  \begin{align*}
  \lambda & \quad \text{(avg # arriving customers/s)} = 10/s \\
  T_{\text{ser}} & \quad \text{(avg time to service customer)} = 20 \text{ ms (0.02s)} \\
  u & \quad \text{(server utilization)} = \lambda \times T_{\text{ser}} = 10/s \times 0.02s = 0.2 \\
  T_q & \quad \text{(avg time/customer in queue)} = T_{\text{ser}} \times u/(1 - u) \\
  & \quad = 20 \times 0.2/(1-0.2) = 20 \times 0.25 = 5 \text{ ms (0.005s)} \\
  L_q & \quad \text{(avg length of queue)} = \lambda \times T_q = 10/s \times 0.005s = 0.05 \\
  T_{\text{sys}} & \quad \text{(avg time/customer in system)} = T_q + T_{\text{ser}} = 25 \text{ ms}
  \end{align*}
\]
Optimize I/O Performance

- How to improve performance?
  - Make everything faster 😊
  - More Decoupled (Parallelism) systems
    » multiple independent buses or controllers
  - Optimize the bottleneck to increase service rate
    » Use the queue to optimize the service
  - Do other useful work while waiting

- Queues absorb bursts and smooth the flow
- Admissions control (finite queues)
  - Limits delays, but may introduce unfairness and livelock

Response Time = Queue + I/O device service time
When is the disk performance highest

- When there are big sequential reads, or
- When there is so much work to do that they can be piggy backed (c-scan)

- OK, to be inefficient when things are mostly idle
- Bursts are both a threat and an opportunity
Summary

- Devices have complex protocols for interaction and performance characteristics
  - Response time (Latency) = Queue + Overhead + Transfer
    - Effective BW = BW * T/(S+T)
  - HDD: controller + seek + rotation + transfer
  - SDD: controller + transfer (erasure & wear)
- Bursts & High Utilization introduce queuing delays
- Systems (e.g., file system) designed to optimize performance and reliability
  - Relative to performance characteristics of underlying device
- Disk Performance:
  - Queuing time + Controller + Seek + Rotational + Transfer
  - Rotational latency: on average ½ rotation
  - Transfer time: spec of disk depends on rotation speed and bit storage density
- Queuing Latency:
  - M/M/1 and M/G/1 queues: simplest to analyze
  - As utilization approaches 100%, latency → ∞
  \[ T_q = T_{ser} \times \frac{1}{2}(1+C) \times \frac{u}{(1 - u)} \]