These slides are mostly taken verbatim, or with minor changes, from those prepared by
Mary Jane Irwin (www.cse.psu.edu/~mji)
of The Pennsylvania State University
[Adapted from Computer Organization and Design, 4th Edition,
Patterson & Hennessy, © 2008, MK]
Key to the Slides

- The source of each slide is coded in the footer on the right side:
  - Irwin CSE331 = slide by Mary Jane Irwin from the course CSE331 (Computer Organization and Design) at The Pennsylvania State University.
  - Irwin CSE431 = slide by Mary Jane Irwin from the course CSE431 (Computer Architecture) at The Pennsylvania State University.
  - Hegner UU = slide by Stephen J. Hegner at Umeå University.
Review: Major Components of a Computer

- Processor
  - Control
  - Datapath

- Memory
  - Main Memory
  - Secondary Memory (Disk)

- Devices
  - Output
  - Input

- Cache
Magnetic Disk

- **Purpose**
  - Long term, *nonvolatile* storage
  - Lowest level in the memory hierarchy
    - slow, large, inexpensive

- **General structure**
  - A rotating platter coated with a magnetic surface
  - A moveable read/write head to access the information on the disk

- **Typical numbers**
  - 1 to 4 platters (each with 2 recordable surfaces) per disk of 1” to 3.5” in diameter
  - Rotational speeds of 5,400 to 15,000 RPM
  - 10,000 to 50,000 *tracks* per surface
    - *cylinder* - all the tracks under the head at a given point on all surfaces
  - 100 to 500 *sectors* per track
    - the smallest unit that can be read/written (typically 512B)
Magnetic Disk Characteristic

- Disk read/write components

  1. **Seek time**: position the head over the proper track (3 to 13 ms avg)
     - due to locality of disk references, the actual average seek time may be only 25% to 33% of the advertised number
  
  2. **Rotational latency**: wait for the desired sector to rotate under the head (½ of 1/RPM converted to ms)
     - 0.5/5400RPM = 5.6ms to 0.5/15000RPM = 2.0ms
  
  3. **Transfer time**: transfer a block of bits (one or more sectors) under the head to the disk controller’s cache (70 to 125 MB/s are typical disk transfer rates in 2008)
     - the disk controller’s “cache” takes advantage of spatial locality in disk accesses
     - cache transfer rates are much faster (e.g., 375 MB/s)
  
  4. **Controller time**: the overhead the disk controller imposes in performing a disk I/O access (typically < .2 ms)
Typical Disk Access Time

- The average time to read or write a 512B sector for a disk rotating at 15,000 RPM with average seek time of 4 ms, a 100MB/sec transfer rate, and a 0.2 ms controller overhead

  \[
  \text{Avg disk read/write} = 4.0 \text{ ms} + \frac{0.5}{(15,000 \text{ RPM}/(60 \text{ sec}/\text{min}))} + \frac{0.5 \text{ KB}}{(100 \text{ MB}/\text{sec})} + 0.2 \text{ ms} = 4.0 + 2.0 + 0.01 + 0.2 = 6.2 \text{ ms}
  \]

  If the measured average seek time is 25% of the advertised average seek time, then

  \[
  \text{Avg disk read/write} = 1.0 + 2.0 + 0.01 + 0.2 = 3.2 \text{ ms}
  \]

- The rotational latency is usually the largest component of the access time
Disk Interface Standards

- Higher-level disk interfaces have a microprocessor disk controller that can lead to performance optimizations
  - ATA (Advanced Technology Attachment) – An interface standard for the connection of storage devices such as hard disks, solid-state drives, and CD-ROM drives. Parallel ATA (PATA) has been largely replaced by serial ATA (SATA).
  - SCSI (Small Computer Systems Interface) – A set of standards (commands, protocols, and electrical and optical interfaces) for physically connecting and transferring data between computers and peripheral devices. Most commonly used for hard disks and tape drives.

- In particular, disk controllers have SRAM disk caches which support fast access to data that was recently read and often also include prefetch algorithms to try to anticipate demand.
### Magnetic Disk Examples ([www.seagate.com](http://www.seagate.com))

<table>
<thead>
<tr>
<th>Feature</th>
<th>Seagate ST31000340NS</th>
<th>Seagate ST973451SS</th>
<th>Seagate ST9160821AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk diameter (inches)</td>
<td>3.5</td>
<td>2.5</td>
<td>2.5</td>
</tr>
<tr>
<td>Capacity (GB)</td>
<td>1000</td>
<td>73</td>
<td>160</td>
</tr>
<tr>
<td># of surfaces (heads)</td>
<td>4</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Rotation speed (RPM)</td>
<td>7,200</td>
<td>15,000</td>
<td>5,400</td>
</tr>
<tr>
<td>Transfer rate (MB/sec)</td>
<td>105</td>
<td>79-112</td>
<td>44</td>
</tr>
<tr>
<td>Minimum seek (ms)</td>
<td>0.8r-1.0w</td>
<td>0.2r-0.4w</td>
<td>1.5r-2.0w</td>
</tr>
<tr>
<td>Average seek (ms)</td>
<td>8.5r-9.5w</td>
<td>2.9r-3.3w</td>
<td>12.5r-13.0w</td>
</tr>
<tr>
<td>MTTF (hours@25°C)</td>
<td>1,200,000</td>
<td>1,600,000</td>
<td>??</td>
</tr>
<tr>
<td>Dim (inches), Weight (lbs)</td>
<td>1x4x5.8, 1.4</td>
<td>0.6x2.8x3.9, 0.5</td>
<td>0.4x2.8x3.9, 0.2</td>
</tr>
<tr>
<td>GB/cu.inch, GB/watt</td>
<td>43, 91</td>
<td>11, 9</td>
<td>37, 84</td>
</tr>
<tr>
<td>Power: op/idle/sb (watts)</td>
<td>11/8/1</td>
<td>8/5.8/-</td>
<td>1.9/0.6/0.2</td>
</tr>
<tr>
<td>Price in 2008, $/GB</td>
<td>~$0.3/GB</td>
<td>~$5/GB</td>
<td>~$0.6/GB</td>
</tr>
</tbody>
</table>
Disk Latency & Bandwidth Milestones

<table>
<thead>
<tr>
<th></th>
<th>CDC Wren</th>
<th>SG ST41</th>
<th>SG ST15</th>
<th>SG ST39</th>
<th>SG ST37</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSpeed (RPM)</td>
<td>3600</td>
<td>5400</td>
<td>7200</td>
<td>10000</td>
<td>15000</td>
</tr>
<tr>
<td>Capacity (Gbytes)</td>
<td>0.03</td>
<td>1.4</td>
<td>4.3</td>
<td>9.1</td>
<td>73.4</td>
</tr>
<tr>
<td>Diameter (inches)</td>
<td>5.25</td>
<td>5.25</td>
<td>3.5</td>
<td>3.0</td>
<td>2.5</td>
</tr>
<tr>
<td>Interface</td>
<td>ST-412</td>
<td>SCSI</td>
<td>SCSI</td>
<td>SCSI</td>
<td>SCSI</td>
</tr>
<tr>
<td>Bandwidth (MB/s)</td>
<td>0.6</td>
<td>4</td>
<td>9</td>
<td>24</td>
<td>86</td>
</tr>
<tr>
<td>Latency (msec)</td>
<td>48.3</td>
<td>17.1</td>
<td>12.7</td>
<td>8.8</td>
<td>5.7</td>
</tr>
</tbody>
</table>

Patterson, CACM Vol 47, #10, 2004

- Disk latency is one average seek time plus the rotational latency.

- Disk bandwidth is the peak transfer time of formatted data from the media (not from the cache).
Latency & Bandwidth Improvements

- In the time that the disk bandwidth doubles, the latency improves by a factor of only 1.2 to 1.4
Flash Storage

- Flash memory is the first credible challenger to disks. It is semiconductor memory that is nonvolatile like disks, but has latency 100 to 1000 times faster than disk and is smaller, more power efficient, and more shock resistant.
  - In 2008, the price of flash is $4 to $10 per GB or about 2 to 10 times higher than disk and 5 to 10 times lower than DRAM.
  - Flash memory bits wear out (unlike disks and DRAMs), but wear leveling can make it unlikely that the write limits of the flash will be exceeded

<table>
<thead>
<tr>
<th>Feature</th>
<th>Kingston</th>
<th>Transend</th>
<th>RiDATA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacity (GB)</td>
<td>8</td>
<td>16</td>
<td>32</td>
</tr>
<tr>
<td>Bytes/sector</td>
<td>512</td>
<td>512</td>
<td>512</td>
</tr>
<tr>
<td>Transfer rates (MB/sec)</td>
<td>4</td>
<td>20r-18w</td>
<td>68r-50w</td>
</tr>
<tr>
<td>MTTF</td>
<td>&gt;1,000,000</td>
<td>&gt;1,000,000</td>
<td>&gt;4,000,000</td>
</tr>
<tr>
<td>Price (2008)</td>
<td>~ $30</td>
<td>~ $70</td>
<td>~ $300</td>
</tr>
</tbody>
</table>
Dependability, Reliability, Availability

- Reliability – measured by the mean time to failure (MTTF). Service interruption is measured by mean time to repair (MTTR).

- Availability – a measure of service accomplishment
  \[ \text{Availability} = \frac{\text{MTTF}}{\text{MTTF} + \text{MTTR}} \]

- To increase MTTF, either improve the quality of the components or design the system to continue operating in the presence of faulty components
  1. Fault avoidance: preventing fault occurrence by construction
  2. Fault tolerance: using redundancy to correct or bypass faulty components (hardware)
     - Fault detection versus fault correction
     - Permanent faults versus transient faults
RAIDs: Disk Arrays

Redundant Array of Inexpensive Disks

- Arrays of small and inexpensive disks
  - Increase potential **throughput** by having many disk drives
    - Data is spread over multiple disk
    - Multiple accesses are made to several disks at a time
- **Reliability** is lower than a single disk
- But **availability** can be improved by adding redundant disks (RAID)
  - Lost information can be reconstructed from redundant information
  - **MTTR**: mean time to repair is in the order of hours
  - **MTTF**: mean time to failure of disks is tens of years
RAID: Level 0 (No Redundancy; Striping)

- Multiple smaller disks as opposed to one big disk
  - Spreading the sector over multiple disks – striping – means that multiple blocks can be accessed in parallel increasing the performance
    - A 4 disk system gives four times the throughput of a 1 disk system
  - Same cost as one big disk – assuming 4 small disks cost the same as one big disk

- No redundancy, so what if one disk fails?
  - Failure of one or more disks is more likely as the number of disks in the system increases
RAID: Level 1 (Redundancy via Mirroring)

- Uses twice as many disks as RAID 0 (e.g., 8 smaller disks with the second set of 4 duplicating the first set) so there are always two copies of the data
  - # redundant disks = # of data disks so twice the cost of one big disk
    - writes have to be made to both sets of disks, so writes would be only 1/2 the performance of a RAID 0

- What if one disk fails?
  - If a disk fails, the system just goes to the “mirror” for the data
RAID: Level 0+1 (Striping with Mirroring)

- Combines the best of RAID 0 and RAID 1, data is striped across four disks and mirrored to four disks
  - Four times the throughput (due to striping)
  - # redundant disks = # of data disks  so twice the cost of one big disk
    - writes have to be made to both sets of disks, so writes would be only 1/2 the performance of RAID 0

- What if one disk fails?
  - If a disk fails, the system just goes to the “mirror” for the data
RAID: Level 3 (Bit-Interleaved Parity)

- Cost of higher availability is reduced to $1/N$ where $N$ is the number of disks in a protection group
  - $\#$ redundant disks = $1 \times \#$ of protection groups
    - writes require writing the new data to the data disk as well as computing the parity, meaning reading the other disks, so that the parity disk can be updated

- Can tolerate limited (single) disk failure, since the data can be reconstructed
  - reads require reading all the operational data disks as well as the parity disk to calculate the missing data that was stored on the failed disk
RAID: Level 4 (Block-Interleaved Parity)

- Cost of higher availability still only 1/N but the parity is stored as blocks associated with sets of data blocks
  - Four times the throughput (striping)
  - \# redundant disks = 1 × \# of protection groups
  - Supports “small reads” and “small writes” (reads and writes that go to just one (or a few) data disk in a protection group)
    - by watching which bits change when writing new information, need only to change the corresponding bits on the parity disk
    - the parity disk must be updated on every write, so it is a bottleneck for back-to-back writes

- Can tolerate limited disk failure, since the data can be reconstructed
Small Writes

- **RAID 3 writes**

  New D1 data

  ![Diagram of RAID 3 writes](image)

  3 reads and 2 writes involving all the disks

- **RAID 4 small writes**

  New D1 data

  ![Diagram of RAID 4 small writes](image)

  2 reads and 2 writes involving just two disks
RAID: Level 5 (Distributed Block-Interleaved Parity)

- Cost of higher availability still only 1/N but the parity block can be located on any of the disks so there is no single bottleneck for writes
  - Still four times the throughput (striping)
  - # redundant disks = 1 × # of protection groups
  - Supports “small reads” and “small writes” (reads and writes that go to just one (or a few) data disk in a protection group)
  - Allows multiple simultaneous writes as long as the accompanying parity blocks are not located on the same disk
- Can tolerate *limited* disk failure, since the data can be reconstructed
By distributing parity blocks to all disks, some small writes can be performed in parallel.
Summary

- Four components of disk access time:
  - Seek Time: advertised to be 3 to 14 ms but lower in real systems
  - Rotational Latency: 5.6 ms at 5400 RPM and 2.0 ms at 15000 RPM
  - Transfer Time: 30 to 80 MB/s
  - Controller Time: typically less than .2 ms

- RAIDs can be used to improve availability
  - RAID 1 and RAID 5 – widely used in servers, one estimate is that 80% of disks in servers are RAIDs
  - RAID 0+1 (mirroring) – EMC, Tandem, IBM
  - RAID 3 – Storage Concepts
  - RAID 4 – Network Appliance

- RAIDs have enough redundancy to allow continuous operation, but not hot swamping
Review: Major Components of a Computer

- Processor
  - Control
  - Datapath

- Memory

- Devices
  - Output
  - Input

- Important metrics for an I/O system
  - Performance
  - Expandability
  - Dependability
  - Cost, size, weight
  - Security
A Typical I/O System

Processor

Cache

Interrupts

Memory - I/O Bus

Main Memory

I/O Controller

I/O Controller

I/O Controller

Disk

Disk

Graphics

Network
Input and Output Devices

- I/O devices are incredibly diverse with respect to
  - Behavior – input, output or storage
  - Partner – human or machine
  - Data rate – the peak rate at which data can be transferred between the I/O device and the main memory or processor

<table>
<thead>
<tr>
<th>Device</th>
<th>Behavior</th>
<th>Partner</th>
<th>Data rate (Mb/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard</td>
<td>input</td>
<td>human</td>
<td>0.0001</td>
</tr>
<tr>
<td>Mouse</td>
<td>input</td>
<td>human</td>
<td>0.0038</td>
</tr>
<tr>
<td>Laser printer</td>
<td>output</td>
<td>human</td>
<td>3.2000</td>
</tr>
<tr>
<td>Magnetic disk</td>
<td>storage</td>
<td>machine</td>
<td>800.0000-3000.000</td>
</tr>
<tr>
<td>Graphics display</td>
<td>output</td>
<td>human</td>
<td>800.0000-8000.000</td>
</tr>
<tr>
<td>Network/LAN</td>
<td>input or output</td>
<td>machine</td>
<td>100.0000-10000.000</td>
</tr>
</tbody>
</table>
I/O Performance Measures

- **I/O bandwidth** (throughput) – amount of information that can be input (output) and communicated across an interconnect (e.g., a bus) to the processor/memory (I/O device) per unit time

  1. How much data can we move through the system in a certain time?
  2. How many I/O operations can we do per unit time?

- **I/O response time** (latency) – the total elapsed time to accomplish an input or output operation

  - An especially important performance metric in real-time systems

- Many applications require *both* high throughput and short response times
I/O System Interconnect Issues

- A **bus** is a shared communication link (a single set of wires used to connect multiple subsystems) that needs to support a range of devices with widely varying latencies and data transfer rates
  - **Advantages**
    - Versatile – new devices can be added easily and can be moved between computer systems that use the same bus standard
    - Low cost – a single set of wires is shared in multiple ways
  - **Disadvantages**
    - Creates a communication bottleneck – bus **bandwidth** limits the maximum I/O **throughput**

- The maximum bus speed is largely limited by
  - The **length** of the bus
  - The **number** of devices on the bus
Types of Buses

- Processor-memory bus ("Front Side Bus", proprietary)
  - Short and high speed
  - Matched to the memory system to maximize the memory-processor bandwidth
  - Optimized for cache block transfers

- I/O bus (industry standard, e.g., SCSI, USB, Firewire)
  - Usually is lengthy and slower
  - Needs to accommodate a wide range of I/O devices
  - Use either the processor-memory bus or a backplane bus to connect to memory

- Backplane bus (industry standard, e.g., ATA, PCIeExpress)
  - Allow processor, memory and I/O devices to coexist on a single bus
  - Used as an intermediary bus connecting I/O busses to the processor-memory bus
I/O Transactions

- An I/O transaction is a sequence of operations over the interconnect that includes a request and may include a response either of which may carry data. A transaction is initiated by a single request and may take many individual bus operations. An I/O transaction typically includes two parts:
  1. Sending the address
  2. Receiving or sending the data

- Bus transactions are defined by what they do to memory:
  - A read transaction reads data from memory (to either the processor or an I/O device)
  - A write transaction writes data to the memory (from either the processor or an I/O device)
Synchronous and Asynchronous Buses

- Synchronous bus (e.g., processor-memory buses)
  - Includes a clock in the control lines and has a fixed protocol for communication that is relative to the clock
  - Advantage: involves very little logic and can run very fast
  - Disadvantages:
    - Every device communicating on the bus must use the same clock rate
    - To avoid clock skew, they cannot be long if they are fast

- Asynchronous bus (e.g., I/O buses)
  - It is not clocked, so requires a handshaking protocol and additional control lines (ReadReq, Ack, DataRdy)
  - Advantages:
    - Can accommodate a wide range of devices and device speeds
    - Can be lengthened without worrying about clock skew or synchronization problems
  - Disadvantage: slow(er)
ATA Cable Sizes

- Companies have transitioned from synchronous, parallel wide buses to asynchronous narrow buses
  - Reflection on wires and clock skew makes it difficult to use 16 to 64 parallel wires running at a high clock rate (e.g., ~400MHz) so companies have moved to buses with a few one-way wires running at a very high “clock” rate (~2GHz)

- Serial ATA cables (red) are much thinner than parallel ATA cables (green)
Asynchronous Bus Handshaking Protocol

- Output (read) data from memory to an I/O device

1. Memory sees ReadReq, reads addr from data lines, and raises Ack
2. I/O device sees Ack and releases the ReadReq and data lines
3. Memory sees ReadReq go low and drops Ack
4. When memory has data ready, it places it on data lines and raises DataRdy
5. I/O device sees DataRdy, reads the data from data lines, and raises Ack
6. Memory sees Ack, releases the data lines, and drops DataRdy
7. I/O device sees DataRdy go low and drops Ack
## Key Characteristics of I/O Standards

<table>
<thead>
<tr>
<th></th>
<th>Firewire</th>
<th>USB 2.0/3.0</th>
<th>PCIe</th>
<th>Serial ATA</th>
<th>SA SCSI</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use</strong></td>
<td>External</td>
<td>External</td>
<td>Internal</td>
<td>Internal</td>
<td>External</td>
</tr>
<tr>
<td><strong>Devices per channel</strong></td>
<td>63</td>
<td>127</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td><strong>Max length</strong></td>
<td>4.5 meters</td>
<td>5 meters (USB 2.0)</td>
<td>0.5 meters</td>
<td>1 meter</td>
<td>8 meters</td>
</tr>
<tr>
<td><strong>Data Width</strong></td>
<td>4</td>
<td>2</td>
<td>2 per lane</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td><strong>Peak Bandwidth</strong></td>
<td>50MB/sec (400) 100MB/sec (800)</td>
<td>0.2MB/sec (low) 1.5MB/sec (full) 60MB/sec (high) 600MB/sec (USB 3.0)</td>
<td>250MB/sec per lane (1x) Come as 1x, 2x, 4x, 8x, 16x, 32x</td>
<td>300MB/sec</td>
<td>300MB/sec</td>
</tr>
<tr>
<td><strong>Hot pluggable?</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Depends</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Revised sjh 20131202
A Typical I/O System

Intel Xeon 5300 processor

Memory Controller Hub (north bridge) 5000P

Front Side Bus (1333MHz, 10.5GB/sec)

Main memory DIMMs

FB DDR2 667 (5.3GB/sec)

ESI (2GB/sec)

PCIe 8x (2GB/sec)

Disk

Serial ATA (300MB/sec)

LPC (1MB/sec)

USB ports

USB 2.0 (60MB/sec)

I/O Controller Hub (south bridge) Entreprise South Bridge 2

CD/DVD

Parallel ATA (100MB/sec)

PCI-X bus (1GB/sec)

PCI-X bus (1GB/sec)

PCIe 4x (1GB/sec)

PCIe 4x (1GB/sec)

Keyboard, Mouse, …

PCIe 4x

(1GB/sec)

USB 2.0

(60MB/sec)
Example: The Pentium 4’s Buses

System Bus (“Front Side Bus”):
64b x 800 MHz (6.4GB/s), 533 MHz, or 400 MHz

Memory Controller Hub (“Northbridge”)

Graphics output: 2.0 GB/s

Gbit ethernet: 0.266 GB/s

2 serial ATAs: 150 MB/s

2 parallel ATA: 100 MB/s

I/O Controller Hub (“Southbridge”)

Hub Bus: 8b x 266 MHz

PCI: 32b x 33 MHz

8 USBs: 60 MB/s

DDR SDRAM Main Memory

Intel® Pentium® 4 Processor

82875P MCH

AGP8X

Communication Streaming Architecture/GbE

Dual Independent Serial ATA Ports
10/100 LAN Connect Interface
Legacy ATA 100

Intel® Hub Architecture

Intel® RAID Technology (ICH5R only)

BIOS Supports HT Technology

DDR400/333 SDRAM

DDR

DDR

6 Channel Audio
Hi-Speed USB 2.0 8 Ports

ICH5/ICH5R

PCI

5DV118 20131208 t:6 sl:40
Irwin CSE431 PSU
Interfacing I/O Devices to the Processor, Memory, and OS

- The operating system acts as the interface between the I/O hardware and the program requesting I/O since
  - Multiple programs using the processor share the I/O system
  - I/O systems usually use interrupts which are handled by the OS
  - Low-level control of an I/O device is complex and detailed

- Thus OS must handle interrupts generated by I/O devices and supply routines for low-level I/O device operations, provide equitable access to the shared I/O resources, protect those I/O devices/activities to which a user program doesn’t have access, and schedule I/O requests to enhance system throughput
  - OS must be able to give commands to the I/O devices
  - I/O device must be able to notify the OS about its status
  - Must be able to transfer data between the memory and the I/O device
Communication of I/O Devices and Processor

- How the processor directs the I/O devices
  - Special I/O instructions
    - Must specify both the device and the command
  - Memory-mapped I/O
    - Portions of the high-order memory address space are assigned to each I/O device
    - Read and writes to those memory addresses are interpreted as commands to the I/O devices
    - Load/stores to the I/O address space can only be done by the OS

- How I/O devices communicate with the processor
  - Polling – the processor periodically checks the status of an I/O device (through the OS) to determine its need for service
    - Processor is totally in control – but does all the work
    - Can waste a lot of processor time due to speed differences
  - Interrupt-driven I/O – the I/O device issues an interrupt to indicate that it needs attention
Interrupt Driven I/O

- An I/O interrupt is **asynchronous** wrt instruction execution
  - Is not associated with any instruction so doesn’t prevent any instruction from completing
    - You can pick your own convenient point to handle the interrupt

- With I/O interrupts
  - Need a way to identify the device generating the interrupt
  - Can have different urgencies (so need a way to **prioritize** them)

- Advantages of using interrupts
  - Relieves the processor from having to continuously poll for an I/O event; user program progress is only suspended during the actual transfer of I/O data to/from user memory space

- Disadvantage – special hardware is needed to
  - Indicate the I/O device causing the interrupt and to save the necessary information prior to servicing the interrupt and to resume normal processing after servicing the interrupt
Interrupt Priority Levels

- Priority levels can be used to direct the OS the order in which the interrupts should be serviced
  - MIPS Status register
    - Determines who can interrupt the processor (if Interrupt enable is 0, none can interrupt)
    - Interrupt levels are prioritized from left to right.
  - MIPS Cause register
    - Pending interrupts have occurred but have not yet been serviced.
    - To enable a Pending interrupt, the correspond bit in the Interrupt mask must be 1
    - Once an interrupt occurs, the OS can find the reason in the Exception codes field
Interrupt Handling Steps

1. Interrupt-enable bit of the status register is set to 0.
2. Logically AND the Pending interrupt field and the Interrupt mask field to see which enabled interrupts could be the culprit. Make copies of both Status and Cause registers.
3. Select the higher priority of these interrupts (leftmost is highest)
4. Save the Interrupt mask field
5. Change the Interrupt mask field to disable all interrupts of equal or lower priority
6. Save the processor state prior to “handling” the interrupt
7. Set the Interrupt enable bit to 1 (to allow higher-priority interrupts)
8. Call the appropriate interrupt handler routine
9. Before returning from interrupt, set the Interrupt enable bit back to 0 and restore the Interrupt mask field
10. Interrupt-enable bit of the status register is set back to 1.

Interrupt priority levels (IPLs) assigned by the OS to each process can be raised and lowered via changes to the Status’s Interrupt mask field.
Direct Memory Access (DMA)

- For high-bandwidth devices (like disks) interrupt-driven I/O would consume a lot of processor cycles
- With DMA, the DMA controller has the ability to transfer large blocks of data directly to/from the memory without involving the processor
  1. The processor initiates the DMA transfer by supplying the I/O device address, the operation to be performed, the memory address destination/source, the number of bytes to transfer
  2. The DMA controller manages the entire transfer (possibly thousand of bytes in length), arbitrating for the bus
  3. When the DMA transfer is complete, the DMA controller interrupts the processor to let it know that the transfer is complete
- There may be multiple DMA devices in one system
  - Processor and DMA controllers contend for bus cycles and for memory
The DMA Stale Data Problem

- In systems with caches, there can be two copies of a data item, one in the cache and one in the main memory
  - For a DMA input (from disk to memory) – the processor will be using stale data if that location is also in the cache
  - For a DMA output (from memory to disk) and a write-back cache – the I/O device will receive stale data if the data is in the cache and has not yet been written back to the memory

- The coherency problem can be solved by
  - Routing all I/O activity through the cache – expensive and a large negative performance impact
  - Having the OS invalidate all the entries in the cache for an I/O input or force write-backs for an I/O output (called a cache flush)
  - Providing hardware to selectively invalidate cache entries – i.e., need a snoopinig cache controller
DMA and Virtual Memory Considerations

- Should the DMA work with virtual addresses or physical addresses?

- If working with physical addresses
  - Must constrain all of the DMA transfers to stay within one page because if it crosses a page boundary, then it won’t necessarily be contiguous in memory
  - If the transfer won’t fit in a single page, it can be broken into a series of transfers (each of which fit in a page) which are handled individually and chained together

- If working with virtual addresses
  - The DMA controller will have to translate the virtual address to a physical address (i.e., will need a TLB structure)

- Whichever is used, the OS must cooperate by not remapping pages while a DMA transfer involving that page is in progress
I/O System Performance

- Designing an I/O system to meet a set of bandwidth and/or latency constraints means

1. Finding the weakest link in the I/O system – the component that constrains the design
   - The processor and memory system?
   - The underlying interconnection (i.e., bus)?
   - The I/O controllers?
   - The I/O devices themselves?

2. (Re)configuring the weakest link to meet the bandwidth and/or latency requirements

3. Determining requirements for the rest of the components and (re)configuring them to support this latency and/or bandwidth
I/O System Performance Example

- A disk workload consisting of 64KB reads and writes where the user program executes 200,000 instructions per disk I/O operation and
  - a processor that sustains 3 billion instr/s and averages 100,000 OS instructions to handle a disk I/O operation

The maximum disk I/O rate (# I/O’s/sec) of the processor is

- a memory-I/O bus that sustains a transfer rate of 1000 MB/s
  Each disk I/O reads/writes 64 KB so the maximum I/O rate of the bus is

- SCSI disk I/O controllers with a DMA transfer rate of 320 MB/s that can accommodate up to 7 disks per controller
- disk drives with a read/write bandwidth of 75 MB/s and an average seek plus rotational latency of 6 ms

what is the maximum sustainable I/O rate and what is the number of disks and SCSI controllers required to achieve that rate?
I/O System Performance Example

- A disk workload consisting of 64KB reads and writes where the user program executes 200,000 instructions per disk I/O operation and
  - a processor that sustains 3 billion instr/s and averages 100,000 OS instructions to handle a disk I/O operation
- a memory-I/O bus that sustains a transfer rate of 1000 MB/s
- SCSI disk I/O controllers with a DMA transfer rate of 320 MB/s that can accommodate up to 7 disks per controller
- disk drives with a read/write bandwidth of 75 MB/s and an average seek plus rotational latency of 6 ms

What is the maximum sustainable I/O rate and what is the number of disks and SCSI controllers required to achieve that rate?

The maximum disk I/O rate (# I/O’s/s) of the processor is

\[
\frac{3 \times 10^9}{(200 + 100) \times 10^3} = 10,000 \text{ I/O’s/s}
\]

Each disk I/O reads/writes 64 KB so the maximum I/O rate of the bus is

\[
\frac{1000 \times 10^6}{64 \times 10^3} = 15,625 \text{ I/O’s/s}
\]
Disk I/O System Example

Processor

Cache

Memory - I/O Bus

Main Memory

I/O Controller

Disk ... Disk

Up to 7

320 MB/s

I/O Controller

Disk ... Disk

10,000 I/O’s/s

15,625 I/O’s/s

75 MB/s
I/O System Performance Example, Con’t

So the processor is the bottleneck, not the bus

- disk drives with a read/write bandwidth of 75 MB/s and an average seek plus rotational latency of 6 ms

Disk I/O read/write time = seek + rotational time + transfer time =
6ms + 64KB/(75MB/s) = 6.9ms

Thus each disk can complete 1000ms/6.9ms or 146 I/O’s per second. To saturate the processor requires 10,000 I/O’s per second or
10,000/146 = 69 disks

To calculate the number of SCSI disk controllers, we need to know the average transfer rate per disk to ensure we can put the maximum of 7 disks per SCSI controller and that a disk controller won’t saturate the memory-I/O bus during a DMA transfer

Disk transfer rate = (transfer size)/(transfer time) = 64KB/6.9ms = 9.56 MB/s

Thus 7 disks won’t saturate either the SCSI controller (with a maximum transfer rate of 320 MB/s) or the memory-I/O bus (1000 MB/s). This means we will need 69/7 or 10 SCSI controllers.