ECE/CS 552: Intro to Computer Architecture

Instructor: Mikko Lipasti
TA: Guangyu Shi
Input/Output

• Motivation
• Disks
• Networks
• Buses
• Interfaces
• Examples
Input/Output

• I/O necessary
  – To/from users (display, keyboard, mouse)
  – To/from non-volatile media (disk, tape)
  – To/from other computers (networks)

• Key questions
  – How fast?
  – Getting faster?
Typical Collection of I/O Devices

Figure 6.1
## Examples

<table>
<thead>
<tr>
<th>Device</th>
<th>I or O?</th>
<th>Partner</th>
<th>Data Rate KB/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mouse</td>
<td>I</td>
<td>Human</td>
<td>0.01</td>
</tr>
<tr>
<td>Display</td>
<td>O</td>
<td>Human</td>
<td>60,000</td>
</tr>
<tr>
<td>Modem</td>
<td>I/O</td>
<td>Machine</td>
<td>2-8</td>
</tr>
<tr>
<td>LAN</td>
<td>I/O</td>
<td>Machine</td>
<td>10,000</td>
</tr>
<tr>
<td>Tape</td>
<td>Storage</td>
<td>Machine</td>
<td>2000</td>
</tr>
<tr>
<td>Disk</td>
<td>Storage</td>
<td>Machine</td>
<td>2000-100,000</td>
</tr>
</tbody>
</table>
I/O Performance

• What is performance?
• Supercomputers read/write 1GB of data
  – Want high bandwidth to vast data (bytes/sec)
• Transaction processing does many independent small I/Os
  – Want high I/O rates (I/Os per sec)
  – May want fast response times
• File systems
  – Want fast response time first
  – Lots of locality
Magnetic Disks

- Stack of platters
- Two surfaces per platter
- Tracks
- Heads move together
- Sectors
- Disk access
  - Queueing + seek
  - Rotation + transfer
Magnetic Disks

• How it works.

http://www.youtube.com/watch?v=9eMWG3fwiEU&feature=related

http://www.youtube.com/watch?v=_NLW4sU3DN0
Disk Trends

• Disk trends
  – $/MB down (well below $1/GB)
  – Disk diameter: 14” => 3.5” => 1.8” => 1”
  – Seek time down
  – Rotation speed increasing at high end
    • 5400rpm => 7200rpm => 10Krpm => 15Krpm
    • Slower when energy-constrained (laptop, iPod)
  – Transfer rates up
  – Capacity per platter way up (100%/year)
  – Hence, op/s/MB way down
    • High op/s demand forces excess capacity
Flash Storage

• Flash memory
  – A type of EEPROM

• possible substitute of disk
  – Nonvolatile
  – 100-1000 times faster than disks
  – Small, power efficient & shock resistant

• Popular in mobile devices
Flash Storage

• Disadvantage: wear out
  – Not so popular for desktop and servers

• Solution: wear leveling
  – one block with a specially extended life of 100,000+ cycles (regular: ~1000 cycles)
  – erasures and re-writes are distributed evenly across the medium
### Types of Storage

<table>
<thead>
<tr>
<th>Type</th>
<th>Size</th>
<th>Speed</th>
<th>Cost/bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register</td>
<td>&lt; 1KB</td>
<td>&lt; 1ns</td>
<td>$$$$$</td>
</tr>
<tr>
<td>On-chip SRAM</td>
<td>8KB-6MB</td>
<td>&lt; 10ns</td>
<td>$$$</td>
</tr>
<tr>
<td>Off-chip SRAM</td>
<td>1Mb – 16Mb</td>
<td>&lt; 20ns</td>
<td>$$</td>
</tr>
<tr>
<td>DRAM</td>
<td>64MB – 1TB</td>
<td>&lt; 100ns</td>
<td>$</td>
</tr>
<tr>
<td>Flash</td>
<td>64MB – 32GB</td>
<td>&lt; 100us</td>
<td>~c</td>
</tr>
<tr>
<td>Disk</td>
<td>40GB – 1PB</td>
<td>&lt; 20ms</td>
<td>~0</td>
</tr>
</tbody>
</table>

© 2005 Mikko Lipasti
RAID: Redundant Array of Inexpensive Disks

• Dependability
  – Reliability: Measured by MTTF (mean time to failure)
  – Service interruption: measured by MTTR (mean time to repair)
  – Availability = MTTF/(MTTF + MTTR)

• What if we need 100 disks for storage?
  • MTTF = 5 years / 100 = 18 days!

• RAID 0
  – Data striped (spread over multiple disks), but no error protection

• RAID 1
  – Mirror = stored twice = 100% overhead (most expensive)

• RAID 5
  – Block-wise parity = small overhead and small writes
RAID 0: Block-level stripping

RAID 0

Disk 0
A1
A2
A3
A4
A5
A5
A6
A7
A8

Disk 1
RAID 1: Mirroring

Disk 0

Disk 1
RAID 2: Bit-level Interleave with ECC
RAID 3: Byte-level Interleave

RAID 3

Disk 0
A1, A4, B1, B4

Disk 1
A2, A5, B2, B5

Disk 2
A3, A6, B3, B6

Disk 3
A_p(1-3), A_p(4-6), B_p(1-3), B_p(4-6)

http://en.wikipedia.org
RAID 4: Block-level Interleave

Disk 0

A1
B1
C1
D1

Disk 1

A2
B2
C2
D2

Disk 2

A3
B3
C3
D3

Disk 3

A_p
B_p
C_p
D_p

http://en.wikipedia.org
RAID 5: Block-level interleaved distributed parity

RAID 5

Disk 0

Disk 1

Disk 2

Disk 3

http://en.wikipedia.org
RAID Illustrations

- Raid 0: Stripping
- Raid 1: Mirroring
- Raid 2: Bit ECC
- Raid 3: Byte interleaved
- Raid 4: Block interleaved
- Raid 5 distributed: Block interleaved
- Raid 6

Check disks

Figure 6.12
Tape

• Helical scan
  – 8mm video tape + ECC
  – 7GB/tape at $6/tape = <$1/GB
    • Note similar to cheap IDE hard drives!
  – Tape robots

• E.g. Library of Congress is 10TB text
  – 1500 tapes x $6 = $9000
  – Of course, not that simple
• Helical scan
  – 8mm video tape
  – 7GB/tape at $6/tape = <$1/GB
• Note similar to cheap IDE hard drives!
• Tape robots
  – E.g. Library of Congress is 10TB
  – Tape robots
• 1500 tapes
  – Of course, not that simple
Frame Buffer

• Extreme bandwidth requirement
  – 1560x1280 pixels x 24bits/pixel = 5.7MB
  – Refresh whole screen 30 times/sec = 170MB/s > PCI

• On memory bus
  – Use 24 video DRAMs (dual ported)
    • Refresh display and allow image change by CPU
    • DRAM port
    • Serial port to video

• Also AGP (Accelerated Graphics Port)
  – Video card talks directly to DRAM (not thru PCI)
LAN = Ethernet

• Original Ethernet
  – One-write bus with collisions and exponential back-off
  – Within building
  – 10Mb/s (~= 1MB/s)
• Now Ethernet is
  – Point to point clients (switched network)
  – Client s/w, protocol unchanged
  – 100Mb/s => 1Gb/s
LAN

- Ethernet not technically optimal
  - 80x86 is not technically optimal either!
- Nevertheless, many efforts to displace it have failed (token ring, ATM)
- Emerging: System Area Network (SAN)
  - Reduce SW stack (TCP/IP processing)
  - Reduce HW stack (interface on memory bus)
  - Standard: Infiniband (http://www.infinibandta.org)
Bus

- A shared communication link between processor and memory, and I/O devices
- Consisting of control lines and data lines
  - Control: determine which device gets to access the bus, what type of information on data lines
  - Data: address, command, data to and from devices
- Function:
  - Access control, arbitration: right to use bus
  - Ensure safe transaction in asynchronous transfer using hand-shaking protocol, and/or ECC
Buses

• Bunch of wires
  – Arbitration
  – Control/command
  – Data
  – Address
  – Flexible, low cost
  – Can be bandwidth bottleneck

• Types
  – Processor-memory
    • Short, fast, custom
  – I/O
    • Long, slow, standard
  – Backplane
    • Medium, medium, standard
Buses in a Computer System
Buses

• Synchronous – has clock
  – Everyone watches clock and latches at appropriate phase
  – Transactions take fixed or variable number of clocks
  – Faster but clock limits length
  – E.g. processor-memory

• Asynchronous – requires handshake
  – More flexible
  – I/O
  – Handshaking protocol: A series of steps used to coordinate asynchronous bus transfers in which the sender and receiver proceed to the next step only when both parties agree that the current step has been completed.
Async. Handshake (Fig. 8.10)

Orange: I/O device, Black: Memory

(1) Request made by I/O device & (2) ack send by Memory
(3) Request deasserted & (4) ack deasserted
Wait till memory has data ready, it raise data ready and place data on data bus
(5) Data sent & (6) Data rec’d by I/O device and raise ack line & (7) ack deasserted and data off line by memory
Buses

• Improving bandwidth
  – Wider bus
  – Separate/multiplexed address/data lines
  – Block transfer
    • Spatial locality
Bus Arbitration

- Resolving bus control conflicts and assigning priorities to the requests for control of the bus.
- One or more bus masters, others slaves
  - Bus request
  - Bus grant
  - Priority
  - Fairness
- Implementations
  - Centralized (Arbiter, can be part of CPU or separate device)
  - Distributed (e.g. Ethernet)
Bus Mastering

- Allows bus to communicate directly with other devices on the bus without going through CPU

- Devices capable to take control of the bus.

- Frees up the processor (to do other work simultaneously)
Buses

• Bus architecture / standards
  – ISA (Industry Standard Architecture)
  – MCA (Micro Channel Architecture)
  – EISA (Extended Industry Standard Architecture)
  – VLB (Vesa Local Bus)
  – PCI (Peripheral Communications Interconnect)

• PCI
  – 32 or 64 bit
  – Synchronous 33MHz or 66MHz clock
  – Multiple masters
  – 111 MB/s peak bandwidth
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")
  - AGP8X: 2.0 GB/s
  - Gbit ethernet: 0.266 GB/s
  - Communication Streaming Architecture/GbE

- I/O Controller Hub ("Southbridge")
  - 2 serial ATAs: 150 MB/s
  - 2 parallel ATA: 100 MB/s
  - Legacy ATA 100
  - 10/100 LAN Connect Interface
  - Dual Independent Serial ATA Ports
  - I/O Controller Hub ("Southbridge")

- Hub Bus: 8b x 266 MHz
  - DDR SDRAM Main Memory
    - DDR: 6.4 GB/s
    - DDR: 6.4 GB/s
    - DDR400/333 SDRAM

- Graphics output: 2.0 GB/s

- Hub Bus: 8b x 266 MHz
  - PCI: 32b x 33 MHz
  - Hi-Speed USB 2.0 8 Ports: 133 MB/s
  - 8 USBs: 60 MB/s

- PCI: 32b x 33 MHz

- BIOS Supports HT Technology

- Intel® RAID Technology (ICH5R only)

Fig 8.11
Interfacing

• Three key characteristics
  – Multiple users share I/O resource
  – Overhead of managing I/O can be high
  – Low-level details of I/O devices are complex

• Three key functions
  – Virtualize resources – protection, scheduling
  – Use interrupts (similar to exceptions)
  – Device drivers
Interfacing to I/O Devices

I/O Device Communication

Control Flow Granularity

Fine-grained (shallow adapters)

Coarse-grained (deep adapters, e.g. channels)

Mechanics of Control Flow

Outbound Control Flow

Programmed I/O

Memory-mapped Control Registers

Inbound Control Flow

Polling

Interrupt-driven

Mechanics of Data Flow

Programmed I/O

Direct Memory Access (DMA)

Software Cache Coherence

Hardware Cache Coherence
Interfacing

• How do you give I/O device a command?
  – Memory-mapped load/store
    • Special addresses not for memory
    • Send commands as data
    • Cacheable?
  – I/O commands
    • Special opcodes
    • Send over I/O bus
Interfacing

• How do I/O devices communicate w/ CPU?
  – Poll on devices
    • Waste CPU cycles
    • Poll only when device active?
  – Interrupts
    • Similar to exceptions, but asynchronous
    • Info in cause register
    • Possibly vectored interrupt handler
Interfacing

• Transfer data
  – Polling and interrupts – by CPU
  – OS transfers data

• Too many interrupts?
  – Use DMA so interrupt only when done
  – Use I/O channel – extra smart DMA engine
    • Offload I/O functions from CPU
Input/Output

Diagram showing the relationship between Proc, Cache, Memory, Proc, Cache, PCI Bridge, Graphics, and SCSI.
Interfacing

• DMA
  – CPU sets up
    • Device ID, operation, memory address, # of bytes
  – DMA
    • Performs actual transfer (arb, buffers, etc.)
  – Interrupt CPU when done

• Typically I/O bus with devices use DMA
  – E.g. hard drive, NIC
Interfacing

• DMA virtual or physical addresses?

• Cross page boundaries within DMA?
  – Virtual
    • Page table entries, provided by OS
  – Physical
    • One page per transfer
    • OS chains the physical addresses

• No page faults in between – lock pages
Interfacing

- Caches and I/O
  - I/O in front of cache – slows CPU
  - I/O behind cache – cache coherence?
  - OS must invalidate/flush cache first before I/O
Interfacing

• Multiprogramming
  – I/O through OS
  – Syscall interface between program and OS
  – OS checks protections, runs device drivers
  – Suspends current process, switches to other
  – I/O interrupt fielded by O/S
  – O/S completes I/O and makes process runnable
  – After interrupt, run next ready process
Multiprogramming

Single User:

- CPU1
- Disk Access
- CPU1
- Think Time

Time-shared:

- CPU1
- Disk Access
- CPU1
- Think Time
- CPU2
- Disk Access
- CPU2
- Think Time
- CPU3
- Disk Access
- CPU3
- Think Time
Summary – I/O

• I/O devices
  – Human interface – keyboard, mouse, display
  – Nonvolatile storage – hard drive, tape
  – Communication – LAN, modem

• Buses
  – Synchronous, asynchronous
  – Custom vs. standard

• Interfacing
  – O/S: protection, virtualization, multiprogramming
  – Interrupts, DMA, cache coherence