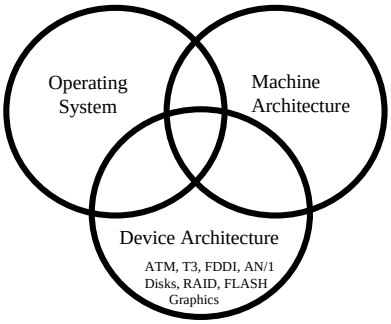
Input/Output Subsystems

- Is I/O important ?
  - Applications executing on the host processor need to be provided with values to operate on, and their results need to be stored away, communicated or displayed
  - Some classes of applications, e.g. multimedia, forecasting, databases, simulations, data mining, distributed systems, servers and graphics rendering, place exacting demands on the I/O subsystem
- Why is now an interesting time ?
  - Application demands are increasing
  - New technologies, e.g. extensible systems, offer more freedom in complete system design
  - Processors went from a phase where their design was an esoteric art to a time when their design is a scientific, quantitative approach, adopting numerous tricks from supercomputing. This may not have fully occurred yet for I/O subsystems
  - Time is ripe for infrastructure upgrades

3/1/99 1

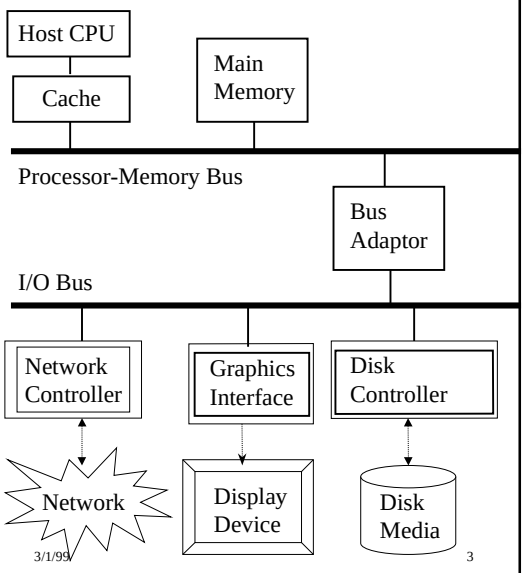
The I/O Challenge



- Bandwidth: Deliver data between the host processor and the I/O devices at maximum device bandwidth
- End-to-end Performance: Turn good host-to-host throughput into equally good application-to-application throughput
- Latency: Provide applications with low latency access to I/O devices
- Protection: Enable applications to safely share I/O devices

3/1/99 2

The Input/Output Subsystem



3/1/99 3

Bus Design

- Definition: Set of wires connecting multiple devices
  - Common access medium
  - Access mediated by a low-level protocol
- Properties of buses:
  - Low cost
  - Compatibility (need only implement one protocol)
  - Easy to add devices (interchangeable)
  - Potentially a bottleneck

3/1/99 4

Bus Design Considerations

- How many buses ? Is one bus good enough ?
  - Complexity+cost versus performance+compatibility tradeoff
- Processor-memory bus
  - High-speed
  - Wide
  - Synchronous protocol
  - Short
  - Tracks processor/memory evolution
- I/O bus
  - Devices of different speeds, different models
  - Narrow
  - Asynchronous
  - Longer
  - Tracks device/controller evolution

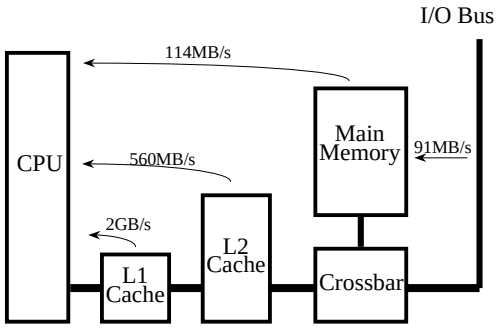
Bus Protocols

- Problem: Buses are shared, therefore are a contended medium
- Need efficient protocols to mediate access
  - Other considerations: Fair, priority based, real-time guarantees
- Bus arbitration:
  - Request, arbitrate, grant, operate, release

Bus Problems & Solutions

- How to reduce protocol overhead ?
  - Lazy release
- How to increase bandwidth ?
  - Wide buses, long transactions
- How to reduce latency, allow sharing, provide timeliness guarantees ?
  - Split transactions
- How to guarantee atomicity ?
  - Synchronization primitives

DEC 3000 (Alpha) Workstation



- Data transfer throughput on various buses
- Common trends among 21164, R10000, Ultra, and P6 in the processor-memory bus:
  - Split transactions
  - Wide: 64-128 bits
  - Xput: 528MB/sec to 1.3GB/sec

Of Mice & Keyboards

- Oldest, simplest device interface
- Based on Programmed I/O (PIO)
  - CPU/device communication over a single address in the host's address space
  - CPU performs all the work, driven by interrupts
  - Peak I/O rate limited by user, therefore bwidth is not an issue

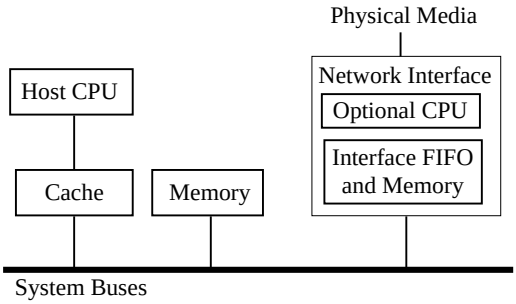
Networking

- Connect computers over various standardized physical communication media
  - Ethernet
  - FDDI
  - ATM
- What are the properties of these communication channels ?
  - Support long distances
  - Lossy

Ethernet

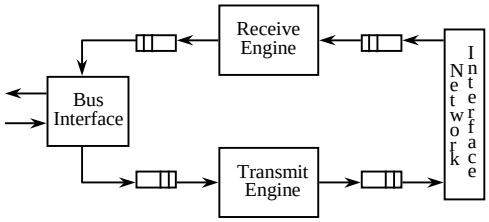
- Shared physical medium
  - "Ether", twisted pair, 10BaseT copper
  - Essentially, a really long, specialized bus
- Serial transmission, 10 Mb/s
- Packet based connectionless network
  - Transactions occur in units of packets, self-contained data that is encapsulated with addressing information
  - SENDTO and BROADCAST transactions
- Distributed arbitration
  - Carrier Sense Multiple Access with Collision Detection (CSMA/CD)
  - Multiple Access: Shared link
  - Carrier Sense: "Listen before talk"
  - Collision Detection: Link-level error detection
- Question: Packets are between 128 and 1530 bytes, why ?

Host Architecture Model



- FIFOs on the interface are used to send and receive network data. Device memory decouples the physical media from the effects of the system bus and memory
- There may be an optional processor on the network interface

Host Interface Design
Simple PIO Interface

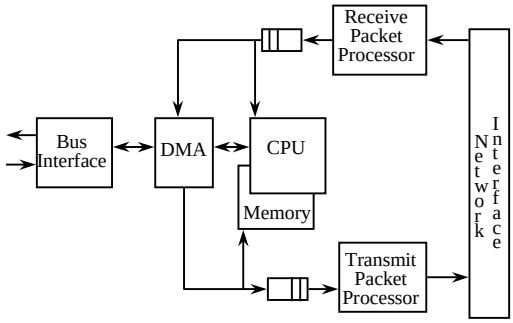


- Programmed I/O FIFOs are accessed via a single address in the host's address space
  - on alpha use memory mapped access and ldq/stq inst
  - on P6 use special I/O instructions
- Interrupts host when data arrives from network
- Advantages
  - reduced implementation complexity & low power budget
  - all protocol processing is done by the host software
- Disadvantages
  - high demand on the host bus and the host CPU
- Example device: Fore ATM 100 series

Direct Memory Access

- Problem: Interrupt driven I/O does not achieve high throughput
  - Peak bandwidth on the bus can be achieved only through burst mode transfer
  - Host CPUs rarely support burst mode transfer
- Solution: place some hardware that transfers data into and out of the device independently of the host CPU
  - Host CPU sets up the DMA
    - Source address, length, target device and location
  - Transfer occurs in the background
  - Interrupt host only when transfer is complete
  - Can perform scatter/gather
    - gather: copy 28 bytes from 0xfff0, 12 from 0xf40 to NIC queue #1

Host Interface Design
Intelligent Interface

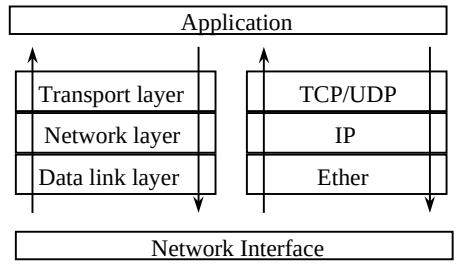


- Intelligent interface with on-board CPU and DMA
  - Processor examples: Intel i/960, ARM, ...
- Local memory is used to store DMA descriptors, code/data for on-board CPU, or for queuing of packets
- Interrupts host when DMA operations complete
- Fore ATM 200 series, Osiris ATM Adapter, Myrinet

Question

- How does data go from the application to the network, and vice-versa ?
  - All access to the network is mediated through the operating system
  - The OS ensures that all packets are properly formatted
  - The OS demultiplexes arriving packets to the intended recipients
- Why ?
  - Current network protocols (TCP/IP) require some trusted code on the

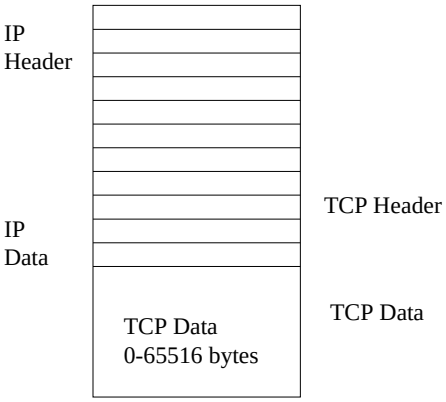
Protocol Basics



- Incoming packets are demultiplexed and given to the intended receiver
- Outgoing packets are encapsulated by lower layer protocol and sent by network interface
- Receive path is asynchronous, and is often the most time consuming part of packet processing

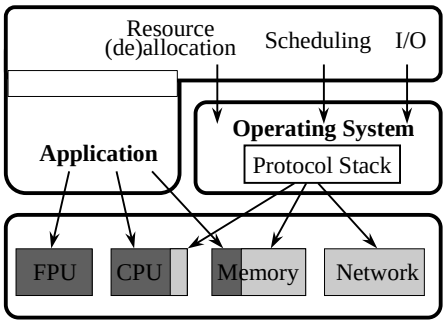
3/1/99 17

Internet Protocols



3/1/99 18

Hardware/Software Layering In Conventional System



Operating System involvement

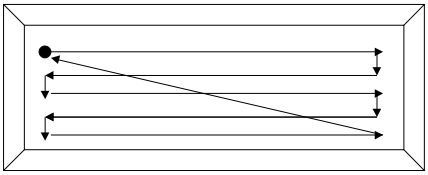
- Resource allocation and deallocation
- Scheduling of interrupts and DMA transfers
- All network I/O

Problems

(and partial solutions)

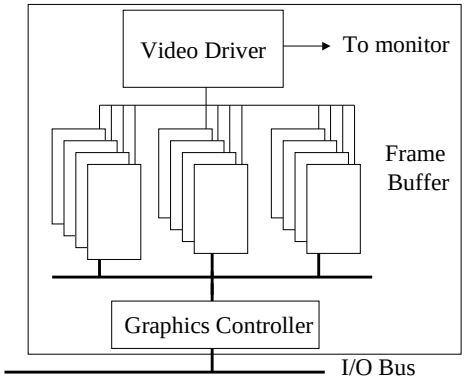
- Short DMA transfers are too costly to set up
  - Use a hybrid interrupt-driven/DMA approach
- Interrupt per packet is too expensive
  - Interrupt only when the receive queue is full, or the transmit queue is half-empty
  - Usepolling
- Applications need to go through the OS for all network access & OS system calls are slow
  - Map device directly into the application's space
  - Download application components into the OS
- Software protocol processing overhead on the host is excessive
  - Offload processing to on-board processor
    - closer to physical network interface
    - higher bandwidth and low latency than host CPU
    - can respond to individual packets more quickly
    - can assist in avoiding host interrupt overhead

Graphics Media



- Cathode ray tube
  - An electron gun scans phosphor particles on a monitor, turning them on by firing when necessary
  - Pixel
  - Bit map
  - Bits per pixel
  - Raster line
  - Refresh rate
  - Vertical blanking interval
- A frame buffer holds the bit map for the screen

Structure of a Typical Graphics Card



- Frame buffer holds the screen image
- Dual-ported DRAMs, a.k.a. video DRAM, is used for the frame buffer (memory-mapped)
- The video generates a raster signal suitable for monitors

Issues in Graphics Card Design

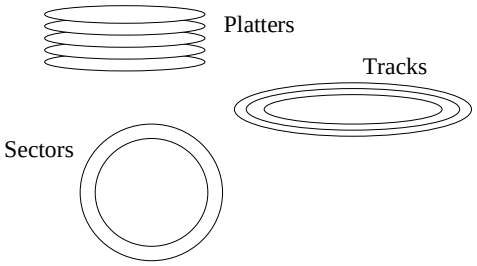
- Support real time requirements for raster generation
  - Dual ported RAM allows memory to be scanned, at high speed, while in use
  - raster bandwidth = Refresh rate * pixel area * depth
- Avoid flicker
  - Modifying the frame buffer while it is being drawn causes flicker
  - Support two frame buffers, write to one frame buffer while displaying the other
- Render three dimensional images in 2D
  - Associate a z axis depth with each pixel, replace a pixel value only if the new value has a smaller z value, i.e. it is closer to the viewer. Cost ?
- Partition screen space between multiple applications
  - O.S. and window manager performs window clipping
  - Cost of OS involvement is large

Issues in Graphics Card Design (2)

- Partitioning screen between applications is costly
  - DirectX is unsafe
  - Solution: Associate an application ID with each pixel. Window manager assigns screen real estate to apps, the graphics controller discards operations to portions of the screen that an application does not own.
  - Cost?
- How much functionality should reside on the card ? What is the interface between the card and the host ?
  - Direct mapped frame buffers force all graphics operations to be performed on the host
  - Trend: High end cards are powerful multiprocessors with abstract (e.g. draw 3D polygon) interfaces

Disks

- Magnetic media
  - Density increases 4X every 3 years (like DRAM)
- Terms: Platter, track, sector, cylinder



- Disk address: cylinder, track, sector

Disk Performance

- How long do accesses take ?
  - Seek time + rotational delay + sector transfer time + controller overhead
- Seek time: 10-20 ms
- Rotational delay = 6-8ms
- Transfer xput = 2-5 MB/sec
- Physical trends: disks are shrinking in size, and rotational speeds are increasing

Disk Controller Interface

- The interface to the disk controller is similar to that of network cards
  - Fundamental advantage: Applications are already written to assume that the disks are a slow medium
  - Interrupt driven: transfers are often large enough that interrupt overhead is not a huge problem
  - DMA: Use burst data transfer to copy data
  - The host interface is high-level and expressive
    - SCSI-2 manual is thicker than the Unix manual
    - Supports device discovery, initialization, reset, query, ordering of operations
- High-level interface enables the controller to be smart behind the scenes
  - All controllers cache at least a sector, often more
  - Controllers perform local scheduling of the disk head
  - Controllers perform error detection and correction
  - Controllers remap bad sectors to create a semi-virtual disk

Disk Controller Problems

- High-level, complex interface has some disadvantages
- The overhead of communication with the controller is large
- The host needs to run a complicated driver to talk to the controller protocol
- Device is too complex to share across applications through simple hardware mechanisms, requires OS as a mediator
- Disks are difficult to model accurately
- The data is far from the host processor, where the computation is performed
  - May have to cross the system I/O bus twice for a simple operation, such as disk sort
  - Can application functionality be migrated closer to the disk head ?

Disks vs. Other I/O

- Every so often, someone forecasts the end of disks, and their replacement by RAM
  - Economics (cost & pricing) has not allowed this to happen, probably will not in the next 10 years
- Disks have mechanical limitations, especially on their write rates, whereas networks do not
  - Projects have looked at how clusters of workstations can act as a storage medium (e.g. GMS)
  - Issues: data stability, failure model

I/O Trends

- Three different media: networks, disks, graphics, three similar designs
- I/O devices share common problems, and exhibit common trends
  - Greater autonomy, less interference with the host processor
  - Evolution towards high-level, abstract interfaces
  - Effort to take the OS out of the way, either through hardware or software means
- Questions for the future: Do standard I/O buses make sense in a world where machines are replaced wholesale every three years ? Can I/O devices be integrated directly with processors ? Can devices support sufficient computational power to perform interesting tasks ?