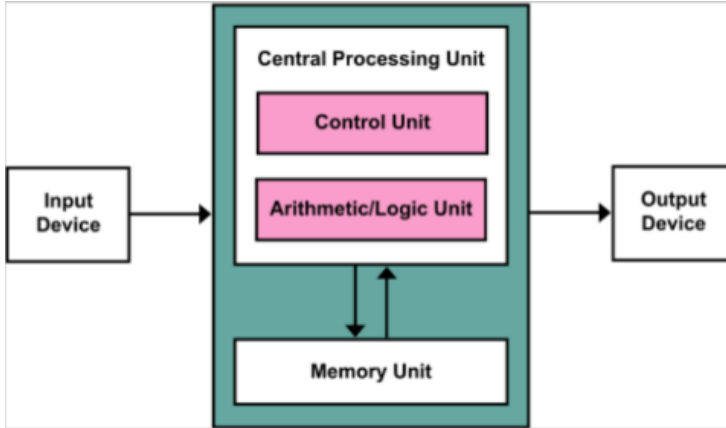
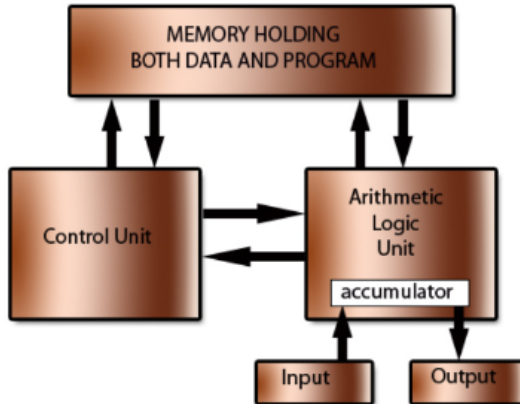


COM1103 Computer Systems

Lecture 1: Overview of Computer Systems	
Computer generations	<p>First-generation computers (1945-1956)</p> <ul style="list-style-type: none">• Vacuum tubes - an electronic component that,<ul style="list-style-type: none">◦ Very inefficient at work;◦ Requires a large cooling system.• Example: ENIAC (Electronic Numeric Integrator And Calculator) <p>Second-generation computers (1956-1963)</p> <ul style="list-style-type: none">• Transistors - replacement of vacuum tubes.<ul style="list-style-type: none">◦ Much smaller in size.• Examples: IBM, Burroughs <p>Third-generation computers (1963-1971)</p> <ul style="list-style-type: none">• IC (Integrated circuits)<ul style="list-style-type: none">◦ Developed the field of microelectronics.◦ Superior performance and reliability (storage capacity and calculating speed).• Semiconductor - a single chip that contains multiple electronic components.• Operating system - a central program that monitors and coordinates the computer's memory for running multiple programs at once. <p>Fourth-generation computers (1971-present)</p> <ul style="list-style-type: none">• "Microprocessor" - a concept that encourages LSI (large scale integration), VLSI (very large scale integration) and ULSI (ultra-large scale integration). <p>Fifth-generation computers (present-future)</p> <ul style="list-style-type: none">• AI (artificial intelligence) - capable of taking self decisions like a human being.
Definition of a computer	<p>Computer</p> <ul style="list-style-type: none">• hardware: a device capable of performing computations and making logical decisions.• software: process data under the control of sets of instructions called computer programs. <p>Hardware</p> <ul style="list-style-type: none">• Consists of various components, including:<ul style="list-style-type: none">◦ CPU (central processing unit)◦ Memory◦ Motherboard◦ Hard disks◦ Peripheral devices (keyboard, screen, mouse CD-ROM)

	<ul style="list-style-type: none"> Every 1-2 years the RAM (memory amount to execute programs) and CPU speed (processor speed) double.
Von Neumann Architecture	<p>Architecture scheme:</p>  <p>Bottleneck:</p> <ul style="list-style-type: none"> Limited throughput (data transfer rate) between CPU and memory, compared to the amount of memory. Reason: the shared bus between the program memory and data memory. 
Computer Organization	<p>Six logical units in every computer</p> <ul style="list-style-type: none"> Input unit - obtains information from input devices, e.g., keyboard, mouse. Output unit - outputs information, e.g., to screen, to print or to control other devices. Memory unit (primary memory) - stores input information with rapid access and low capacity storage. ALU (arithmetic and logic unit) - performs arithmetic calculations and logic decisions. CPU (central processing unit) <ul style="list-style-type: none"> supervises and coordinates the other sections of the computer. later integrates the ALU as a fundamental block.

- Secondary storage unit - stores inactive programs and information with cheap, long-term and high-capacity storage.

Architecture components

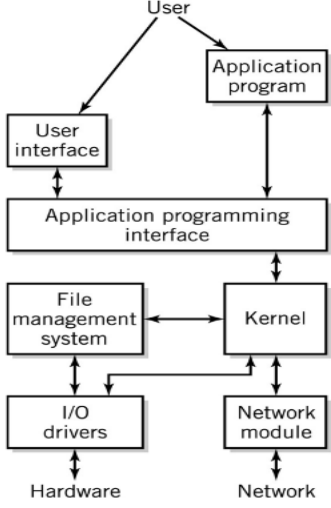
- Hardware
 - Processes data by executing instructions.
 - Provides input and output.
- Software - instructions executed by the system.
- Data - fundamental representation of facts and observations.
- Communications - sharing data and processing among different systems.

Hardware components

- Input/Output devices
- Storage devices
- CPU
 - ALU - performs arithmetic and boolean logical calculations.
 - CU (control unit)
 - Controls processing of instructions.
 - Controls movement of data within the CPU.
 - Interface unit
 - Moves instructions and data between the CPU and other components.
 - Bus - bundle of wires carrying signals and power between different components.
- Memory - short-term storage for CPU calculations.
 - also called “primary storage”, “working storage” and “RAM (random access memory)”.
 - consists of bits, each holding a value of either 0 or 1 (8 bits = 1 byte).

Software components:

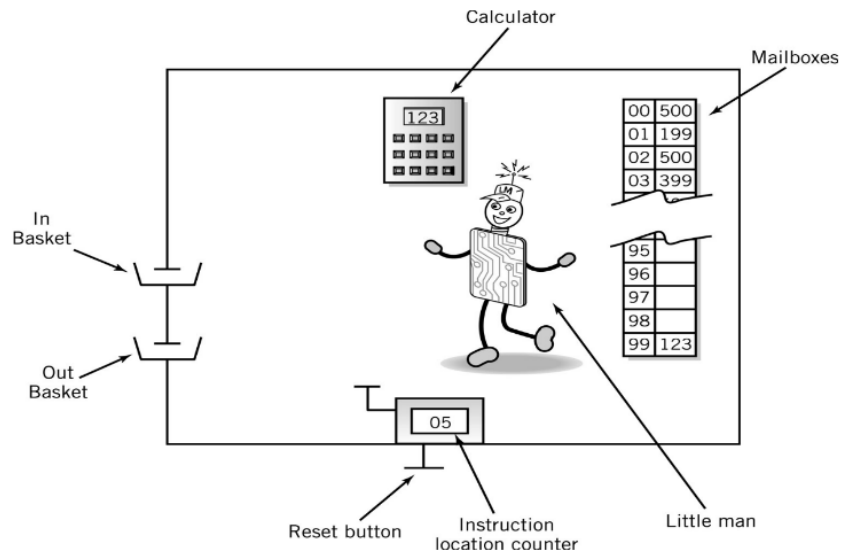
- Applications
- Operating System
 - API (Application Program Interface)
 - File management
 - I/O
 - Kernel (inc. memory management, resource scheduling, program communication, security)
 - Network module
- Evolution of Operating System
 - Batch processing - do one job/task at a time.
 - Operating systems - can manage transitions between jobs.
 - Multiprogramming - shared resources between many jobs/tasks.

	<ul style="list-style-type: none"> ○ Timesharing - can run a small portion and move on to service the next user.  <pre> graph TD User --> UI[User interface] User --> AP[Application program] UI <--> API[Application programming interface] AP <--> API API <--> FMS[File management system] API <--> K[Kernel] FMS <--> K FMS <--> ID[I/O drivers] K <--> NM[Network module] ID <--> H[Hardware] NM <--> N[Network] </pre>
Computer Communications	<p>Communications Component</p> <ul style="list-style-type: none"> ● Hardware <ul style="list-style-type: none"> ○ Communication channels - physical connections between computer systems. <ul style="list-style-type: none"> ■ Examples: Wire cables, phone lines, radio waves, etc. ○ Interface hardware - can handle communication between the computer and the channel. <ul style="list-style-type: none"> ■ Examples: Modem, NIC (Network Interface Card). ● Software <ul style="list-style-type: none"> ○ Network protocols - common ground rules of communication between computers, I/O devices and many software programs. <ul style="list-style-type: none"> ■ Examples: <ul style="list-style-type: none"> ● HTTP - Between Web servers and Web browsers. ● TCP/IP - Between computers on the Internet and local area networks. ● ATAPI - Between a CPU and CD-ROMs. ○ Standards - created to ensure universal compatibility of data formats and protocols. <ul style="list-style-type: none"> ■ By committee or through de facto popular use. ■ Examples: <ul style="list-style-type: none"> ● Computer languages - Java, SQL, C, JavaScript ● Display standards - Postscript, MPEG-2, JPEG, GIF

	<ul style="list-style-type: none"> • Character set standards - ASCII, Unicode • Video standards: VGA, XGA, RGB
Different Computing System	<p>Different computing system</p> <ul style="list-style-type: none"> • Personal computers • Distributed computing - Computing distributed over networks. • Client/server computing - Sharing of information across computer networks between file servers and clients (personal computers). • Mobile computing • Cloud computing

Lecture 1: Little Man Computer and Input/Output Devices

The Little Man Computer




Mailboxes: Address vs. Content

- Addresses - two digits, consecutive.
- Content - three digits, might be data or instructions.
 - Instructions:
 - Op code- operation code or arbitrary mnemonic (short character sequence).
 - Operand - data or address to be manipulated.

Calculator

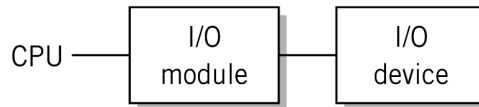
- Three digits.
- Enter and temporarily hold numbers.
- Add and subtract operations.

	<p>Instruction Location Counter</p> <ul style="list-style-type: none"> • Two digits. • Click to increment the count. <ul style="list-style-type: none"> ◦ Little Man can modify the value directly. • Reset button outside. <p>In Basket and Out Basket</p> <ul style="list-style-type: none"> • A slip of paper with a three-digit number. • The only way to communicate with outside, despite the reset button in Instruction Location Counter. <ul style="list-style-type: none"> ◦ LM can write and leave a paper in the Out Basket. ◦ Outside part can put a paper in the In Basket. <p>Assembly Language</p> <ul style="list-style-type: none"> • Specific to a CPU. • 1 to 1 correspondence between its instruction and binary language instruction, represented in Mnemonics. • Examples: <ul style="list-style-type: none"> ◦ Arithmetic: <ul style="list-style-type: none"> ■ Add - 1xx ■ Subtract - 2xx ◦ Data Movement: <ul style="list-style-type: none"> ■ Store - 3xx ■ Load - 5xx ◦ Input/Output: <ul style="list-style-type: none"> ■ Input - 901 ■ Output - 902 ◦ Machine Control(Coffee Break): Stop - 000
I/O Operations	<p>Basic Model of IO:</p> <p>Input → Process → Output</p>  <pre> graph LR Input[Input] --> Process[Process] Process --> Output[Output] </pre> <p>I/O Considerations</p> <ul style="list-style-type: none"> • Speed Issues <ul style="list-style-type: none"> ◦ Devices operate at different speeds. <ul style="list-style-type: none"> ■ The CPU operates much faster than any I/O device. ◦ Burst of data and block data transfer for some devices. • Coordination <ul style="list-style-type: none"> ◦ Several devices perform I/O simultaneously. ◦ Unexpected input and various input formats. ◦ Required status information for devices. <p>I/O Device Interface Issues</p>

- Different formats: parallel interface & serial interface.
- Buffering of data (Data rate related)
- Block Burst v.s. Stream
- Different control requirement (Electromechanical-related)

Simple I/O Configuration:

- Special Interface is needed for providing addressing, synchronization, status and external control capabilities.
 - Different requirements for different devices.



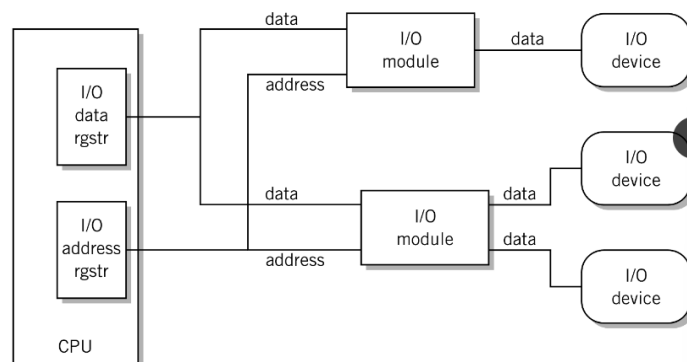
I/O Modules Functions

- Recognizes corresponding messages and accepts commands from the CPU.
 - Provide registers and controls to perform direct memory transfers.
- Physically controls the device.
 - Notifies with interrupts.

Input/Output Modules

- Programmed I/O (CPU controlled I/O)
 - I/O data and address registers in CPU.
 - One word transfers.
 - Address information for each I/O device.
 - Full instruction fetch/execute cycle.
 - Primary use: Keyboards.
- Interrupt Driven I/O
- Direct Memory Access Controllers = method to data transfer from main memory and a device directly without the CPU.

Programmed I/O (CPU controlled I/O):

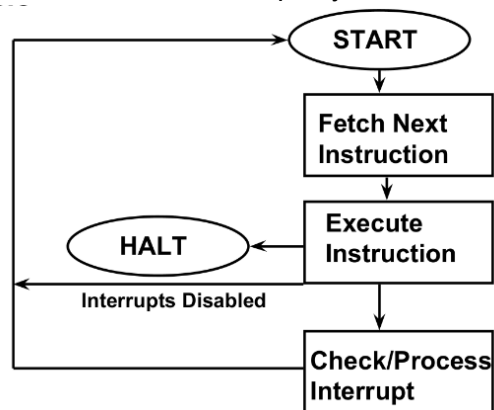


Interrupts

Interrupts

- Signal that causes the CPU to alter its normal flow on instruction execution.
 - Frees CPU from waiting for events.
 - Provides control for external input.
- Examples:
 - Unexpected input (e.g., Control-Alt-Delete).
 - Abnormal situation (e.g., power failure).
 - Illegal instructions (e.g., divide by 0).
 - Multitasking & multiprocessing.

The CPU - The Interrupt Cycle:



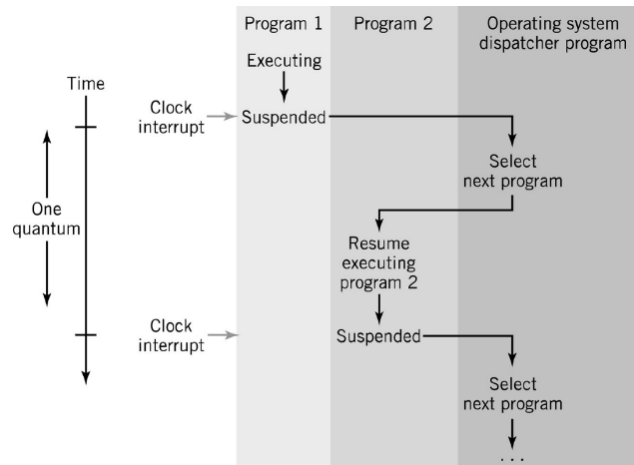
Interrupt Terminology

- Interrupt lines (hardware)
- Interrupt request
- Interrupt handlers (interrupt routine)
 - Program that responds to interrupts.
 - Suspend the program in process.
 - Save information inc. last executed instruction, data values in the PCB.
- PCB (Process Control Block)
 - Located in the stack area (part of the memory).
 - Save all program registers before control transfer to the interrupt handler.

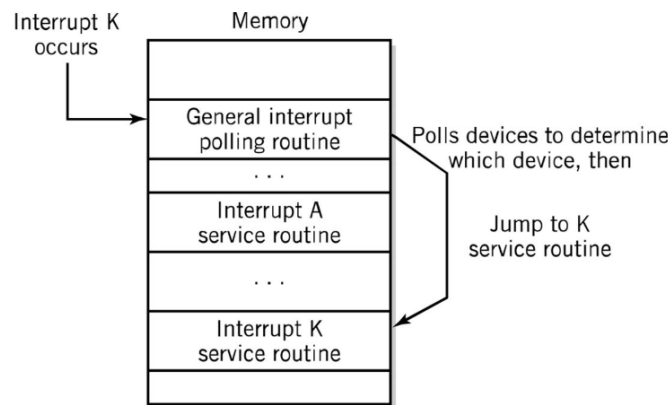
Use of Interrupts

- Notify the occurrence of an external event.
- Signal completion of event, e.g., ready or full.
- Allocate CPU time for time sharing.
- Indicate abnormal event, e.g., illegal operation, hardware error.

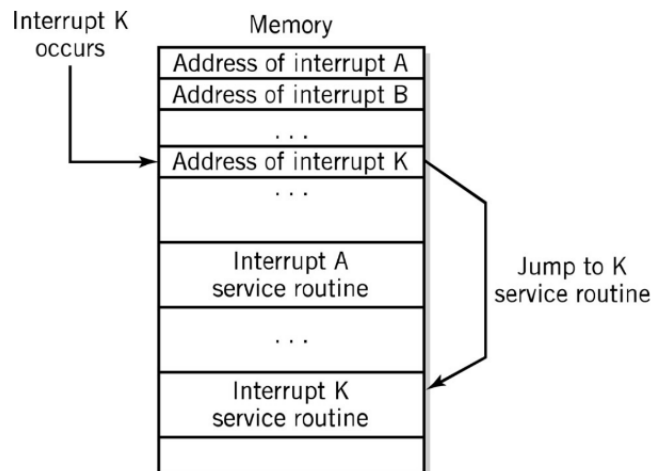
Using an Interrupt for Time Sharing:



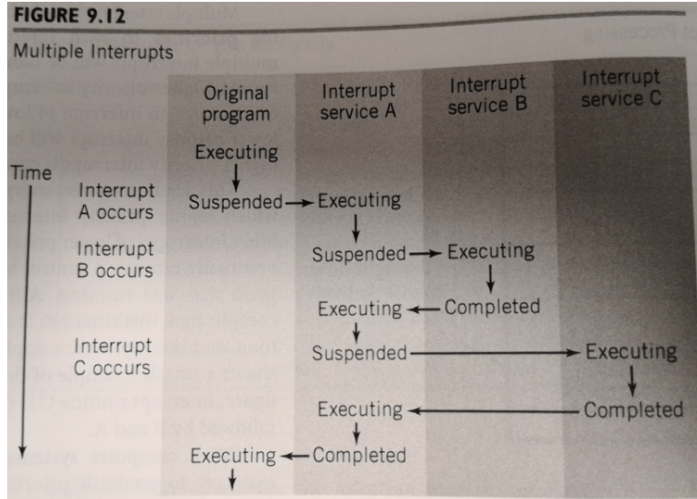
Polled Interrupts:



Vectored Interrupts:



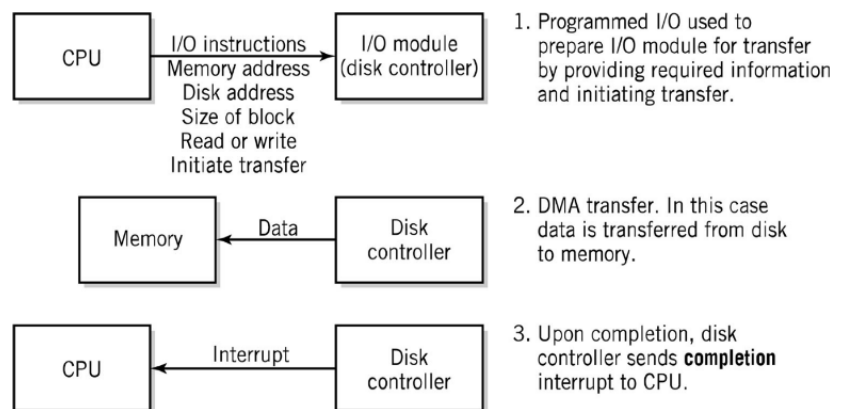
Multiple Interrupts with Different Priorities (Task completion v.s. Loss of data):



DMA Instruction Set

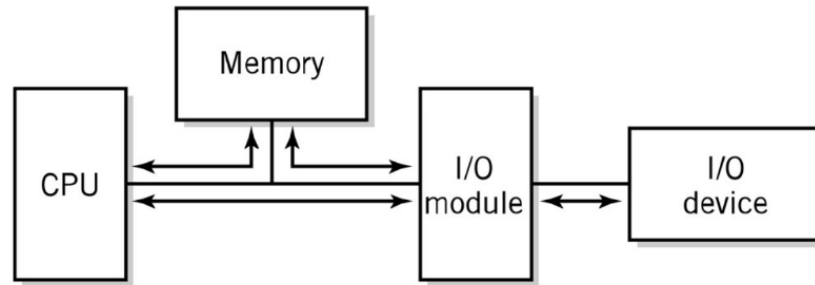
- Privileged instructions = requests of I/O service by application program in operating system.
- Programmed I/O sends the following information to initiate DMA:
 - Location of data on I/O device.
 - Starting location in memory.
 - Size of the block.
 - Read/Write.
- Interrupt the CPU when complete.

DMA Initiation and Control:



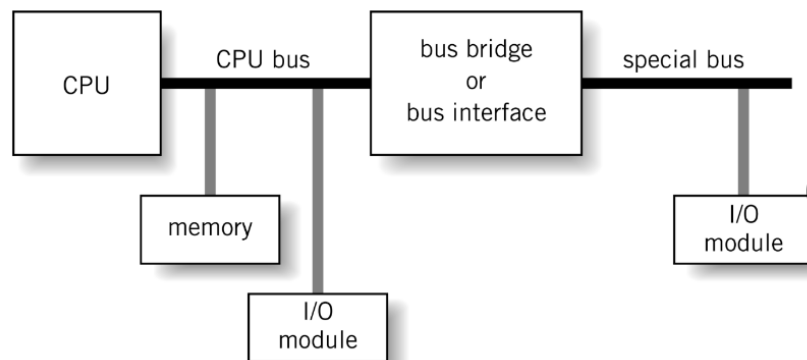
CPU-Memory-I/O Pathway

Basic CPU-Memory-I/O Pathway*:



Source: From *PCI Local Bus Specification Production Version 2*, Copyright © 1993, by PCI Special Interest Group, pg. 9. Reprinted by permission.

Bus Configuration:

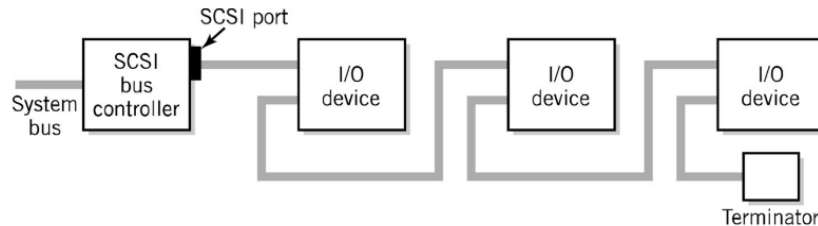


Bus Characteristics

- Carries data width in bits simultaneously.
- Hierarchy:
 - Processor bus - on-chip
 - Cache bus (backside bus)
 - Memory bus (front-side bus) - connect the memory subsystem and processor.
 - Local I/O bus - connect performance critical peripherals to memory and processor at high speed.
 - Examples: PCI, VESA Local Bus.
 - Standard I/O bus - connects slower peripherals (ISA) to local I/O bus.
- External interface buses and ports:
 - Parallel port.
 - Serial port - RS-232C & RS-422 bus.
 - SCSI (Small Computer System Interface)
 - ANSI standard but with multiple variations.
 - An I/O bus that can support multiple devices.
 - USB, USB-2 (Universal Serial Bus)
 - Consists of Hubs that can provide multiple connection points for I/O devices.

- Can support a maximum of 127 devices.
- IEEE 1394 (e.g., Firewire, i.link)

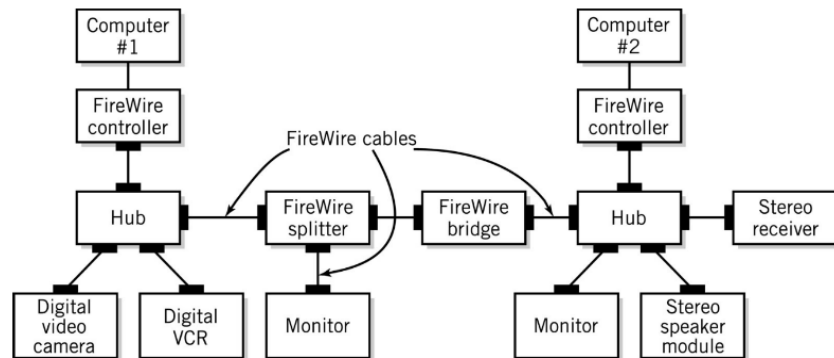
SCSI Bus:



USB vs. FireWire

- Similarities:
 - Both are serial and multipoint bus specifications.
 - Can add/remove devices without powering.
 - Use packet protocol for isochronous data transfer:
 - Deliver at regular time intervals.
 - Guarantee specified throughput.
- Differences:
 - USB = slow to medium speed (12 Mbits/sec) data transfer applications (storage devices).
 - USB-2 = high speed (48 Mbits/sec) data transfer applications.
 - FireWire = even high speed (400 Mbits - 3.2 Gbits / sec) data transfer applications.

Typical FireWire Configuration (Network-like, independent controllers):



Lecture 3: Storage Devices

Peripherals

Peripherals

- Devices that can separate from a basic computer.

- Connect via ports (parallel, USB, serial) or interface to system bus (SCSI, IDE, PCMCIA).
- Are classified into input, output and storage.

Storage Devices

- Primary Storage - Memory
- Secondary Storage:
 - Data and programs must be copied to memory for CPU access.
 - Permanence of data.
 - Include direct access storage Devices (DASDs).
- Access methods:
 - Online storage (burden on bandwidth and network).
 - Offline storage (loaded when needed).

Speed

- Measured by access time and data transfer rate.
 - Access time = average time it takes a computer to locate data and read it.
 - Data transfer rate = amount of moved data per second.
- Measurement unit:
 - Millisecond (ms) = one-thousandth of a second = 10^{-3} s
 - Microsecond (μ s) = 10^{-6} s
 - Nanosecond (ns) = 10^{-9} s
 - Picosecond (ps) = 10^{-12} s

Internal Memory

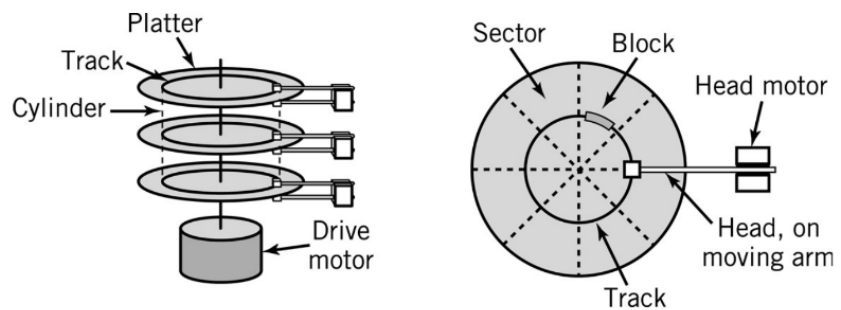
- ROM (Read-only memory): non-volatile, and can retain data without power.
 - EEPROM (Electrically Erasable Programmable ROM)
 - Flash ROM = Faster but more expensive than EEPROM.
 - Usage:
 - BIOS - initial boot instructions and diagnostics (magnetic core memory).
 - Digital cameras.
- RAM:
 - DRAM (Dynamic RAM) = main memory, and has to be refreshed every few milliseconds to retain data.
 - SRAM (Static RAM) = Keeps data in memory as long as with power supply.
 - Faster but more expensive.
 - Frequently used in cache memory for high-speed access (small portion).

Hierarchy of Storage:

<i>Device</i>	<i>Typical Access Times</i>	<i>Throughput Rate</i>
CPU Registers		
Cache Memory (SRAM)	15 to 30 nanoseconds	
Conventional Memory (DRAM)	50 to 100 nanoseconds	
Expanded Storage (RAM)	75 to 500 nanoseconds	
Hard Disk Drive	10 to 50 milliseconds	600 to 6,000 KB/sec
Floppy Disk	95 milliseconds	100 to 200 KB/sec
CD-ROM	100 to 600 milliseconds	500 to 4,000 KB/sec
Tape	.5 and up seconds	2,000 KB/sec (cartridge)

Magnetic Disks

A Hard Disk Layout:



Magnetic Disks

- Structure:
 - Track - circle.
 - Cylinder - same track on all platters.
 - Block - small arc of a track.
 - Sector - pie-shaped part of a platter.
 - Head- reads data off the disk.
- Properties:
 - Same number of bits on each track.
 - Denser number of bits towards the center.
 - CAV = constant angular velocity:
 - Spins the same speed for every track.
 - Hard drives - 3600-7200 rpm
 - Floppy drives - 360 rpm

Locating a Block of Data

- Average seek time = required time to move from one

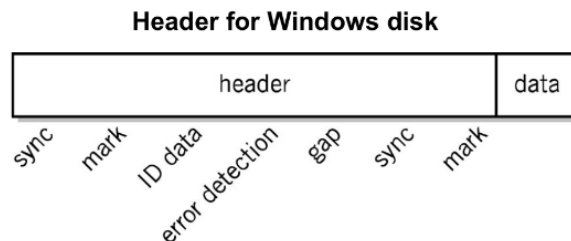
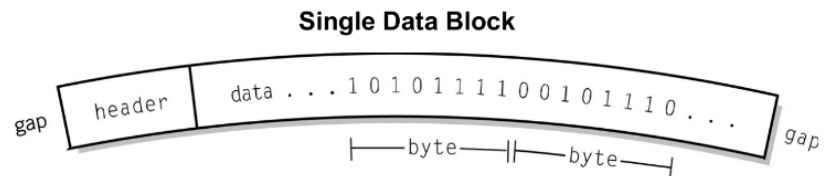
track to another.

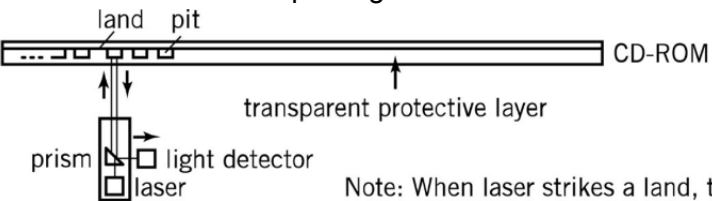
- Latency = required time for the disk to rotate to the beginning of the target sector.
 - Average Latency Time = $1/2 * 1/\text{Rotational Speed}$
- Transfer time = required time to transfer a block of data to the disk controller buffer.
 - $= 1 / (\text{Number of Sectors} * \text{Rotational Speed})$.
- Total time to access a disk block = Average seek time + Average latency time + Average transfer time.

Disk Array RAID - Mirrored & Striped

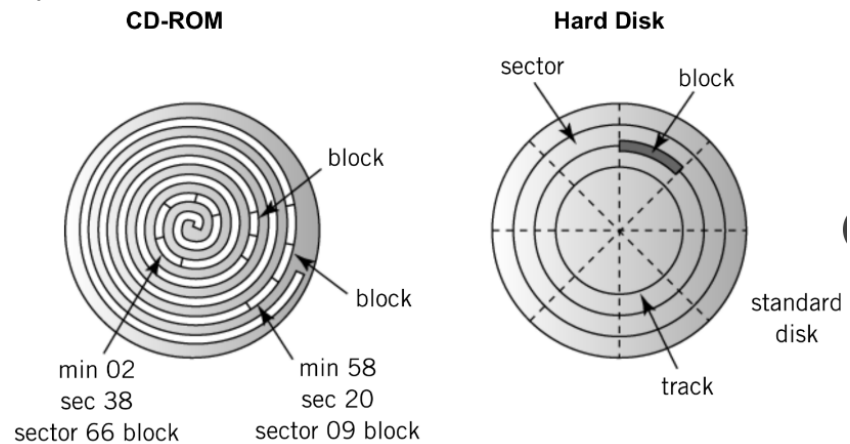
- Mirrored
 - Mirrored array consists of two or more disk drives.
 - Each stores exactly the same data.
 - Advantages:
 - Reduced access time for a multiblock read.
 - Can detect errors by majority logic with three or more drives.
- Striped
 - Request at least three disk drives (one reserved for error checking).
 - File segment is divided into blocks to be stored.
 - Different blocks written to different disks.
 - Parity words are stored on the reserved disk.
 - Advantages:
 - Multiply the throughput rate by the number of data disks in the array.
 - Parity data is used to check the original data during read operations.

Disk Block Formats:



	<p>Alternate Disk Technologies</p> <ul style="list-style-type: none"> • Removable hard drives <ul style="list-style-type: none"> ◦ Disk pack - disk platters are stored in a removable plastic case. <ul style="list-style-type: none"> ■ Or even includes the disk head and arm assembly. • Fixed-head disk drives <ul style="list-style-type: none"> ◦ One head per track. ◦ Can eliminate the seek time. • Bernoulli disk drives <ul style="list-style-type: none"> ◦ Hybrid approach that incorporates both floppy and hard disk technology. ◦ Zip drives. <p>Magnetic Tape</p> <ul style="list-style-type: none"> • Based on archival purposes: <ul style="list-style-type: none"> ◦ Allow offline storage. ◦ Allow disaster recovery. • Tape cartridges: <ul style="list-style-type: none"> ◦ 20 - 144 tracks (side by side). ◦ Read serially (tape backs up). ◦ QIC (Quarter Inch Cartridge) = larger in size. ◦ DAT (Digital Audio Tape) = smaller in size. ◦ Size includes (2:1 compression).
Optical Storage	<p>Optical Storage</p> <ul style="list-style-type: none"> • Reflected light off a mirrored or pitted surface. • CD-ROM <ul style="list-style-type: none"> ◦ Spiral 3 miles long and contains 15 billion bits. ◦ CLV (Constant Linear Velocity) = same physical length among blocks. ◦ Block = 2352 bytes: <ul style="list-style-type: none"> ■ 2k data (2048 bytes). ■ 16 bytes for header (12 starter and 4 id). ■ 288 bytes for advanced error control. • DVD-ROM <ul style="list-style-type: none"> ◦ 4.7G per layer. ◦ Maximum 2 layers per side (each side = 8.5G). • Laser: <ul style="list-style-type: none"> ◦ Strike land - light reflected into detector. ◦ Strike a pit = light scattered.  <p>Note: When laser strikes a land, the light is reflected into the detector; when the light strikes a pit, it is scattered.</p>

Layout: CD-ROM vs. Standard Disk



CD-ROMs:

<i>General Speed</i>	<i>Seek Time (milliseconds)</i>	<i>Data Transfer Rate</i>
Single-Speed	600	150K per second
2X	320	300K per second
3X	250	450K per second
4X	135-180	600K per second
6X	135-180	900K per second
8X	135-180	1.2 MBps
10X	135-180	1.6 MBps
12X	100-150	1.8 MBps
16X	100-150	2.4 MBps (maximum)
24X	100-150	3.6 Mbps (maximum)
32X	100-150	4.8 Mbps (maximum)

Types of Optical Storage

- WORM Disks
 - Require only one write and can read many times.
 - Medium can be altered to blister the surface by using a medium-powered laser.
 - Store data in concentric tracks.
 - Sector data similar to a magnetic disk.
 - CAV (Constant Angular Velocity).
- Medium-powered laser blister technology also used in:
 - CD-R, DVD-R, DVD-ROM.
 - CD-RW, DVD-RW, DVD-RAM, DVD+RAM.
- Magneto-Optical Disks

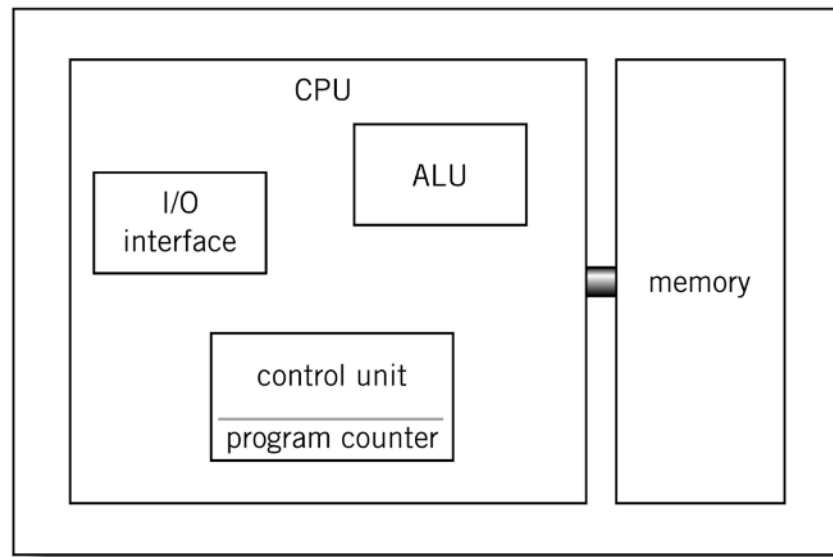
Lecture 4: CPU and Memory

3 Major Components

CPU: 3 Major Components

- ALU (Arithmetic Logic Unit) - performs calculations and comparisons (data change).
- CU (Control Unit) - performs fetch/execute cycle.
 - Functions:
 - Moves data to and from CPU registers and other hardware components (no change in data).
 - Accesses program instruction and issues commands to the ALU.
 - Subparts:
 - Memory Management Unit - supervises fetching instructions and data.
 - I/O Interface - combines Memory Management Unit as Bust Interface Unit sometimes.
- Registers
 - PC (Program Counter) / Instruction Pointer - determines next instruction for execution.

System Block Diagram

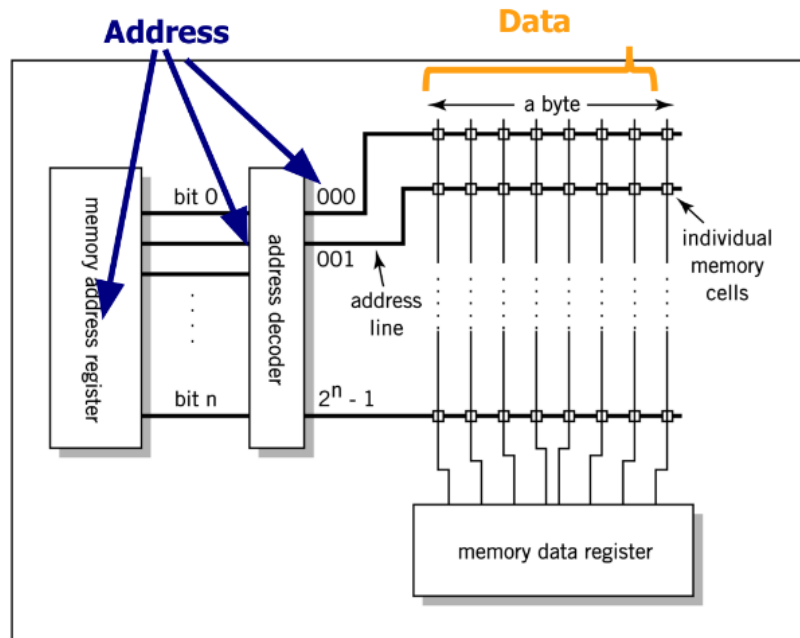


Registers

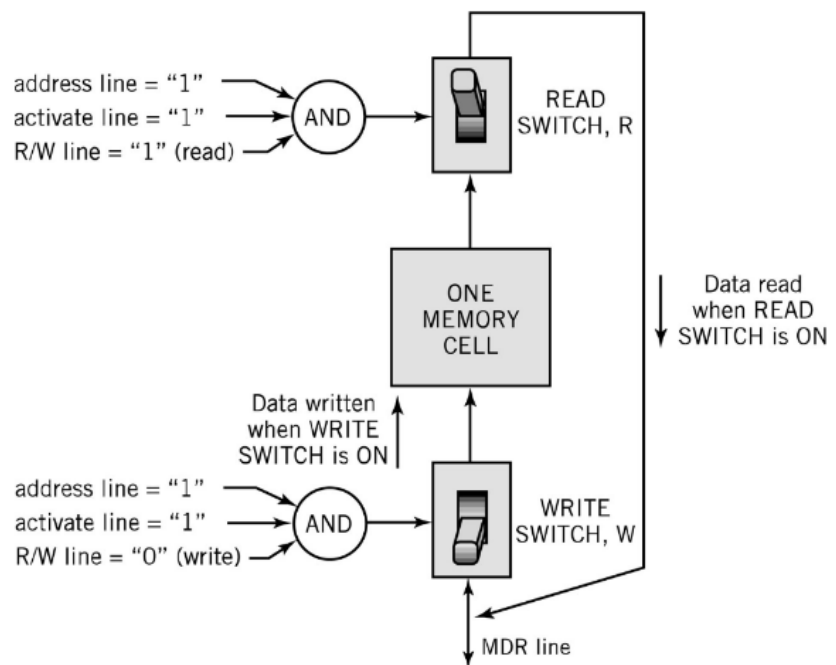
Concept of Registers

- Use small storage locations within the CPU.
 - Size in bits or bytes (different from MB-like memory).
- Be manipulated directly by the CU, wired for specific function.
- Use of Registers:
 - Serve as scratchpad for executing programs.
 - Hold frequently-needed data.

	<ul style="list-style-type: none"> ○ Store information about the status of CPU and executing programs, including: <ul style="list-style-type: none"> ■ Address of the next program instruction. ■ Signals from external devices. <p>General Purpose Registers (User-visible Registers)</p> <ul style="list-style-type: none"> ● Hold intermediate results or data values, e.g., loop counters. <ul style="list-style-type: none"> ○ Equivalent to LMC's calculator. ○ Several dozen contained in current CPUs. <p>Special Purpose Registers</p> <ul style="list-style-type: none"> ● PC (Program Count Register, or Instruction Pointer) ● IR (Instruction Register) - store instruction fetched from memory. ● MAR (Memory Address Register) ● MDR (Memory Data Register) ● Status Registers - store information related to: <ul style="list-style-type: none"> ○ Status of CPU and executing programs. ○ Flags (one bit Boolean variable) that track conditions, including: <ul style="list-style-type: none"> ■ Arithmetic carry and overflow. ■ Power failure. ■ Internal computer error. <p>Register Operations</p> <ul style="list-style-type: none"> ● Store values from registers and the memory. ● Can perform: <ul style="list-style-type: none"> ○ Addition and subtraction. ○ Shift or rotate data. ○ Test conditions with contents, e.g., zero or positive.
The Memory	<p>Operation of Memory</p> <ul style="list-style-type: none"> ● Copy instruction's address to the MAR to find the memory's location. <ul style="list-style-type: none"> ○ Unique address in each memory location. ● The CPU then determines if it is a store or a retrieval, and transfers between the MDR and memory. <ul style="list-style-type: none"> ○ MDR = a two-way register. <p>Relationship between MAR, MDR and Memory:</p>

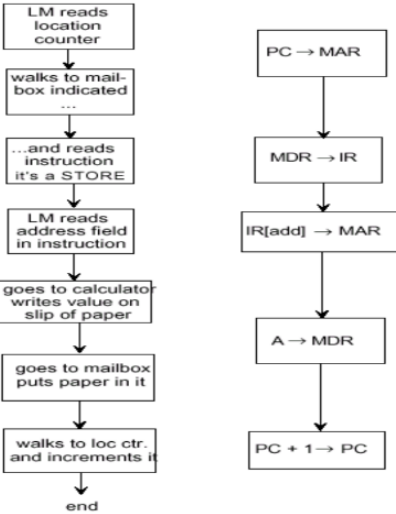


Individual Memory Cell



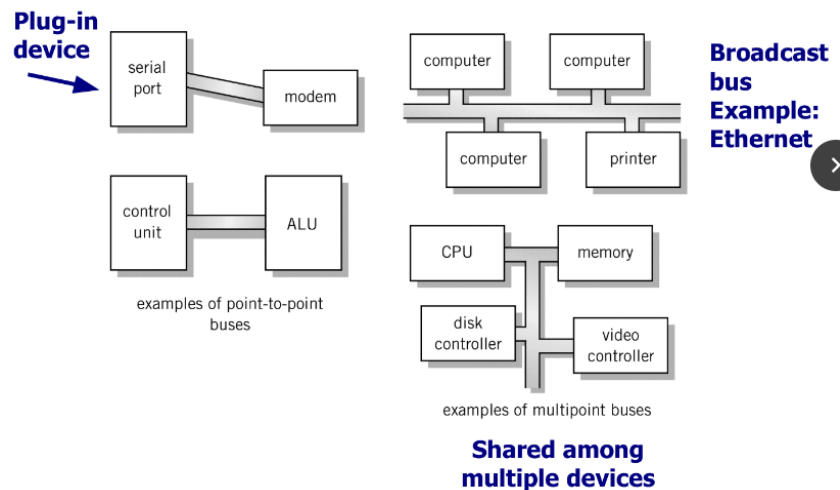
Determinants of Memory Capacity

- Number of bits in the MAR:
 - LMC = 100 (00 to 99)
 - = 2^K allowed locations (K = width of the register in bits).
- Size of the address portion of the instruction.

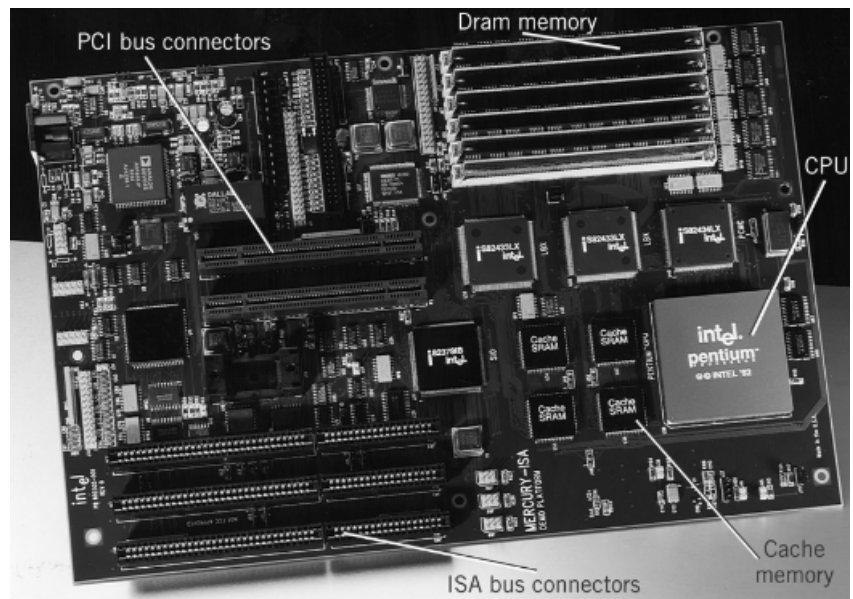
Fetch-Execute Cycle	<p>Fetch-Execute Cycle</p> <ul style="list-style-type: none"> Definition: Two-cycle process as both instructions and data are in memory. Fetch = decode or find instruction from memory into register and signal ALU. Execute = perform operations required by instructions and move/transform data. <p>LMC vs. CPU Fetch and Execute Cycle:</p>  <pre> graph TD LMC[LMC Cycle] LMC1[LM reads location counter] --> LMC2[walks to mailbox indicated ...] LMC2 --> LMC3[...and reads instruction it's a STORE] LMC3 --> LMC4[LM reads address field in instruction] LMC4 --> LMC5[goes to calculator writes value on slip of paper] LMC5 --> LMC6[goes to mailbox puts paper in it] LMC6 --> LMC7[walks to loc ctr. and increments it] LMC7 --> LMC8[end] CPU[CPU Cycle] CPU1[PC → MAR] --> CPU2[MDR → IR] CPU2 --> CPU3[IR[add] → MAR] CPU3 --> CPU4[A → MDR] CPU4 --> CPU5[PC + 1 → PC] </pre> <p>Load Fetch/Execute Cycle</p> <ul style="list-style-type: none"> Transfer the address from PC to MAR. Transfer the instruction from MDR to IR. Address portion of the instruction (IR) in MAR. Copy actual data from MDR to A (Accumulator). Increment PC (Program Counter). <p>Store Fetch/Execute Cycle</p> <ul style="list-style-type: none"> Transfer the address from PC to MAR. Transfer the instruction from MDR to IR. Address portion of the instruction (IR) in MAR. Copy data from A (Accumulator) to MAR. Increment PC (Program Counter). <p>ADD Fetch/Execute Cycle</p> <ul style="list-style-type: none"> Transfer the address from PC to MAR. Transfer the instruction from MDR to IR. Address portion of the instruction (IR) in MR. Add contents of MDR to the contents of A (Accumulator). Increment PC (Program Counter).
Bus	<p>Bus</p> <ul style="list-style-type: none"> Definition = the physical connection to transfer data from one location to another in a computer system.

- Groups of electrical conductors to carry signals.
 - Link = each conductor in the bus.
- 4 kinds of signals:
 - Data (alphanumeric, numerical, instructions)
 - Addresses
 - Control signals
 - Power (sometimes)
- Characteristics:
 - Parallel bus = a bus that supports an individual line for each bit of data, address, and control.
 - All transfer bits on the bus can be transferred simultaneously.
 - Serial bus = a bus that supports a single data line pair to transfer data sequentially, one bit at a time, using a single data line pair.
 - Require a data return line.
 - Point-to-point bus = a bus that connects an external device to a connector (usually cables).
 - Multipoint bus (i.e. broadcast bus) = a bus that produces signals through a source and broadcasts to every other point.
 - Similar to a radio station.
 - Example: traditional Ethernet network.

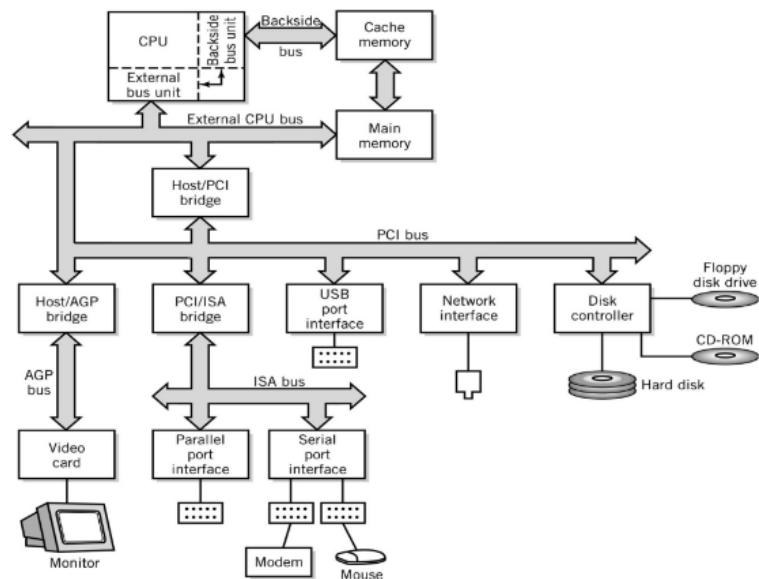
Point-to-point vs. Multipoint:



Motherboard:



Typical PC Interconnections:



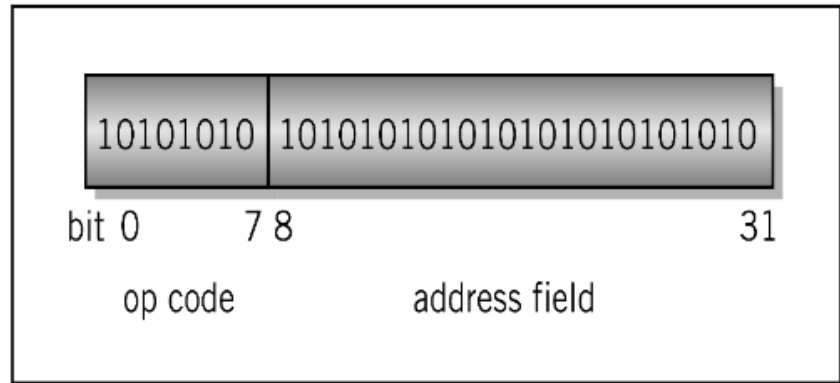
Bus interface bridges connect different bus types

PCI Bus Connections:

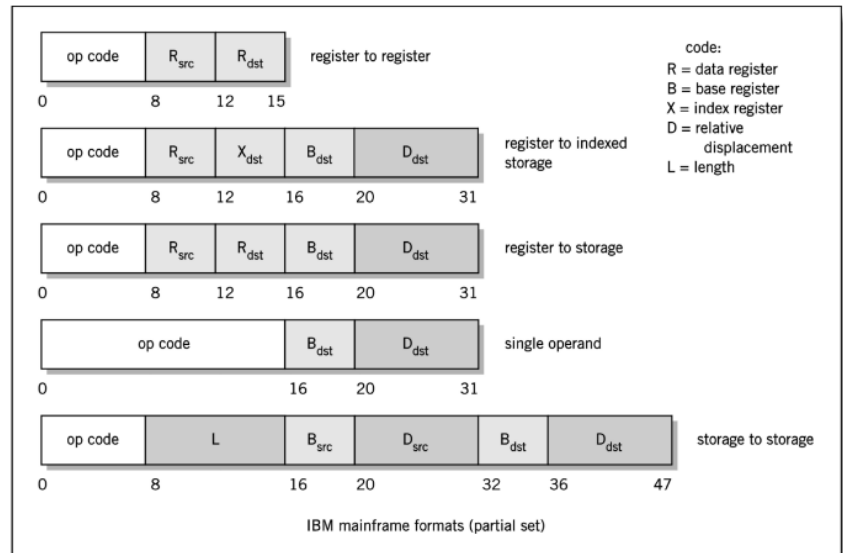
	<div data-bbox="592 226 1409 892"> <p>The diagram illustrates the pin connections for a PCI interface. It is divided into two main sections: Required pins and Optional pins, separated by a central vertical bar representing the PCI bus.</p> <ul style="list-style-type: none"> Required pins: <ul style="list-style-type: none"> Address and data: AD31-00 (input), PAR (bidirectional). PCI command: C/BE3-0 (input). Interface control: FRAME (bidirectional), TRDY (bidirectional), IRDY (bidirectional), STOP (bidirectional), DEVSEL (bidirectional), IDSEL (input). Error reporting: PERR (bidirectional), SERR (bidirectional). Bus arbitration: REQ (input), GNT (input), CLK (input), RST (input). Optional pins: <ul style="list-style-type: none"> Address and data: AD63-32 (input), PAR64 (bidirectional), REQ64 (bidirectional), ACK64 (bidirectional), C/BE7-4 (input). Interface control: LOCK (bidirectional). Interrupts: INTA (input), INTB (input), INTC (input), INTD (input). JTAG test support: TDI (input), TDO (input), TCK (input), TMS (input), TRST (input). <p>Source: Copyright © PCI Pin List/PCI Special Interest Group, 1999.</p> </div>
<p>Instructions</p>	<p>Instruction</p> <ul style="list-style-type: none"> Definition: Directions given to a computer by sending electrical signals through specific circuits. <p>Instruction set</p> <ul style="list-style-type: none"> Defines computer architecture through: <ul style="list-style-type: none"> Number of instructions. Complexity of operations performed by individual instructions. Data types supported. Format (layout, fixed vs. variable length). Use of registers. Addressing (size, modes). <p>Instruction Elements</p> <ul style="list-style-type: none"> OPCODE: task Addresses: <ul style="list-style-type: none"> Source OPERAND(s) Result OPERAND = location of data (register/memory) <ul style="list-style-type: none"> Can be explicit or implicit (default assumed). <p>Instruction Format</p> <ul style="list-style-type: none"> Definition: Machine-specific template that specifies, <ul style="list-style-type: none"> Length of the op code. Number of operands.

- Length of operands.

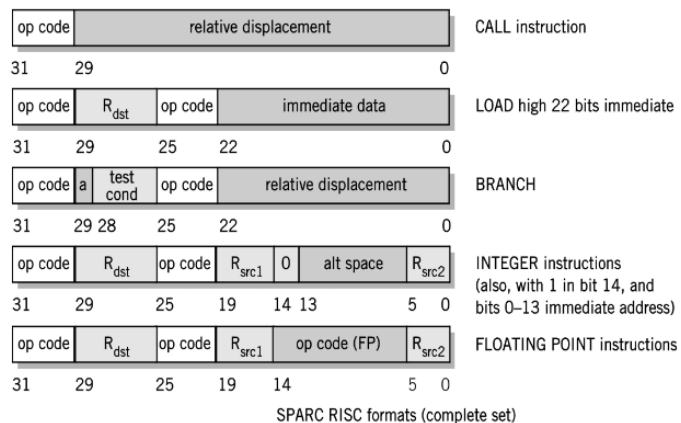
Simple 32-bit Instruction Format:



Instruction Formats: CISC (Complex Instruction Set Computer)



Instruction Formats: RISC (Reduced Instruction Set Computer)



CISC VS RISC

- CISC are usually under-utilized (about 80%).
 - Normally used in Intel x86 CPU.
- ARM (Advanced RISC Machines) is used in smartphones

RISC vs. CISC

CISC	RISC
Emphasis on hardware	Emphasis on software
Multiple instruction sizes and formats	Instructions of same set with few formats
Less registers	Uses more registers
More addressing modes	Fewer addressing modes
Extensive use of microprogramming	Complexity in compiler
Instructions take a varying amount of cycle time	Instructions take one cycle time
Pipelining is difficult	Pipelining is easy

Instruction Types

- Data Transfer (load, store)
 - Most common and have the greatest flexibility.
 - Involve memory and registers.
- Arithmetic = Include operators (+-*^), integers and floating point.
- Logical or Boolean = Include relational operators (> < =) and boolean operators (AND, OR, XOR, NOR and NOT).
- Signal operand manipulation instructions = include negating, decrementing and incrementing.
- More Instruction Types:
 - Bit manipulation instructions = flags to test for

	<p>conditions.</p> <ul style="list-style-type: none"> ○ Shift and rotate. ○ Program control. ○ Stack instructions. ○ Multiple data instructions. ○ I/O and machine control.
--	--

Lecture 5: Number System and Boolean Logic

Number System	<p>Reasons behind using binary</p> <ul style="list-style-type: none"> ● Early computer design was decimal. <ul style="list-style-type: none"> ○ To simplify computer design. ○ Can be used for both instructions and data. ● Natural relationship between on/off switches and calculation using Boolean logic. <p>Counting and Arithmetic</p> <ul style="list-style-type: none"> ● Original (by fingers): base 10 number system. <ul style="list-style-type: none"> ○ Base = the number of different digits in the number system (including zero). ● Categories: <ul style="list-style-type: none"> ○ Binary (Base 2) <ul style="list-style-type: none"> ■ Bit (Binary digit) - 0 and 1. ○ Octal (Base 8) - 0 to 7. ○ Hexadecimal (Base 16) - 0 to F (15). <p>Keeping Track of the Bits</p> <ul style="list-style-type: none"> ● Bits are commonly stored and manipulated in groups (9 bits = 1 byte/word (in many systems)). ● Number of bits: limit size of number manipulated by the computer and affect accuracy of results. <p>Number System</p> <ul style="list-style-type: none"> ● Roman: position independent. ● Modern: based on positional notation. <ul style="list-style-type: none"> ○ Decimal system = system of positional notation based on powers of 10. ○ Binary system = system of positional notation based powers of 2. ○ Octal system = system of positional notation based on powers of 8. ○ Hexadecimal system = system of positional notation based powers of 16 ● Examples: <ul style="list-style-type: none"> ○ $527 \text{ (in base 10)} = 5 * (10^2) + 2 * (10^1) + 7 * (10^0)$ ○ $11010110 \text{ (in base 2)} = 1 * (2^7) + 1 * (2^6) + 0 * (2^5) + 1 * (2^4) + 0 * (2^3) + 1 * (2^2) + 1 * (2^1)$
---------------	--

$$+ 0 * (2^0) = 214 \text{ (in base 10)}$$

Range of Possible Numbers

- $R = B^K$ where
 - R = Range
 - B = Base
 - K = No. of Digits

Decimal Range for Bit Widths:

Bits	Digits	Range
1	0+	2 (0 and 1)
4	1+	16 (0 to 15)
8	2+	256
10	3	1,024 (1K)
16	4+	65,536 (64K)
20	6	1,048,576 (1M)
32	9+	4,294,967,296 (4G)
64	19+	Approx. 1.6×10^{19}
128	38+	Approx. 2.6×10^{38}

Number of Symbols vs. Number of Digits

- For a given number, the larger the base the more symbols required, but the fewer digits needed.

Base 10 Addition Table:

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	3	4	5	6	7	8	9	10	11
3	3	4	5	6	7	8	9	10	11	12
4	4	5	6	7	8	9	10	11	12	13

etc

Base 8 Addition Table:

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

Base 10 Multiplication Table:

x	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70

Base 8 Multiplication Table:

x	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	4	6	10	12	14	16	20
3	3	6	11	14	17	22	25	30
4	4	10	14	20	24	30	34	40
5	5	12	17	24	31	36	43	50
6	6	14	22	30	36	44	52	60
7	7	16	25	34	43	52	61	70

Boolean Logic

Binary Arithmetic

- Addition = Boolean using XOR and AND.
 - XOR = Output is "1" only if either input, but not both inputs, is a "1".
 - AND = Output is "1" if and only both inputs are a "1".
- Multiplication = Boolean using AND and Shift.
 - AND = Output is "1" if and only both inputs are a "1".
 - Shift = Shifting a number in any base left one digit multiplies its value by the base.
- Division = Boolean using AND and Shift.
 - AND = Output is "1" if and only both inputs are a "1".

- Shift = Shifting a number in any base right one digit divides its value by the base.

From Base 10 to Base 2:

Base 10 42

Quotient

$$\begin{array}{r}
 2 \overline{) 42} \text{ (0)} \\
 \underline{21} \\
 2 \overline{) 21} \text{ (1)} \\
 \underline{10} \\
 2 \overline{) 10} \text{ (0)} \\
 \underline{5} \\
 2 \overline{) 5} \text{ (1)} \\
 \underline{2} \\
 2 \overline{) 2} \text{ (0)} \\
 \underline{1} \\
 2 \overline{) 1} \text{ (0)} \\
 \underline{0}
 \end{array}$$

Remainder

Least significant bit

Most significant bit

Base 2 101010

From Base 10 to Base 16:

Base 10 8,039

Quotient

$$\begin{array}{r}
 16 \overline{) 8,039} \text{ (7)} \\
 \underline{112} \\
 16 \overline{) 502} \text{ (6)} \\
 \underline{96} \\
 16 \overline{) 31} \text{ (15)} \\
 \underline{240} \\
 16 \overline{) 1} \text{ (1)} \\
 \underline{16} \\
 16 \overline{) 0}
 \end{array}$$

Remainder

Least significant bit

Most significant bit

Base 16 1F67

From Base 16 to Base 2

- The nibble approach - hex is easier to read and write compared to binary.

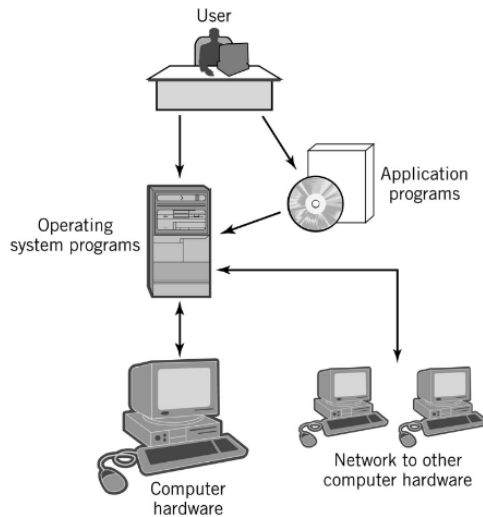
Base 16	1	F	6	7
Base 2	0001	1111	0110	0111

Decimal Fractions

- Move the number point one place to the right = multiple

	<p>the number by the base number.</p> <ul style="list-style-type: none"> • Move the number point one place to the left = divide the number by the base number. <p>Fractions: Base 10 and Base 2</p> <ul style="list-style-type: none"> • No general relationship between fractions of types $1/10^k$ and $1/2^k$. <ul style="list-style-type: none"> ◦ A number representable in base 10 may not be representable in base 2. ◦ All fractions of the form $1/2^k$ can be represented in base 10. <p>Mixed Number Conversion</p> <ul style="list-style-type: none"> • Integer and fraction parts must be converted separately. • Radix point: fixed reference for the conversion. <ul style="list-style-type: none"> ◦ Digit to the left is a unit digit in every base. ◦ B^0 is always 1 regardless of the base.
--	---

Lecture 6: Introduction to Operating System	
Operating System	<p>Barebones Computer System</p> <ul style="list-style-type: none"> • Do not load instructions into the main memory. <ul style="list-style-type: none"> ◦ Remain idle when waiting for user input. ◦ Cannot run multiple programs at once. • No interface except for I/O routines provided with executing programs. <ul style="list-style-type: none"> ◦ No facility to store, retrieve or manipulate files. ◦ No ability to control peripheral devices. <p>Operating System</p> <ul style="list-style-type: none"> • Definition: A collection of computer programs that integrate and make available hardware resources to a user and his/her programs. <ul style="list-style-type: none"> ◦ Allowing a more productive, time-efficient and effective access. <p>Integrated Computer Environment:</p>

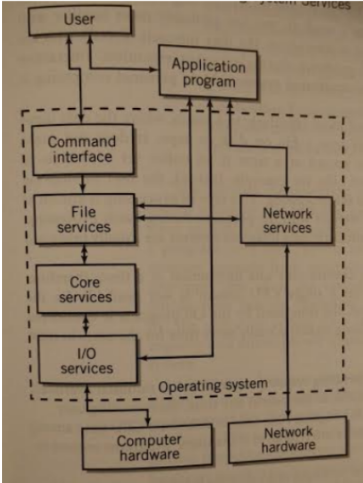


Services of A Operating System

- Basic Services
 - Provide programs to accept commands and requests.
 - Manage, load and execute programs.
 - Manage hardware resources.
 - Act as an interface between the user and the system.
- Advance Services
 - Provide File support services.
 - Provide I/O support services.
 - Provide Network services.
 - Provide concurrent processing tools and services.
 - Allocate resources such as memory, CPU time and I/O devices.
 - Protect users and programs from each other and provide inter-program communication.
 - Provide feedback to system administrators to permit performance optimization.
 - Act as a means to bootstrap or boost the computer.
 - Handle all interrupt processing.

OS Parts

- Memory Resident (the Kernel)
 - Always loaded in memory.
 - Contains essential required services.
 - Typically responsible for managing memory management, processes and tasks, and secondary storage.
- Memory Non-resident (i.e. Applications)

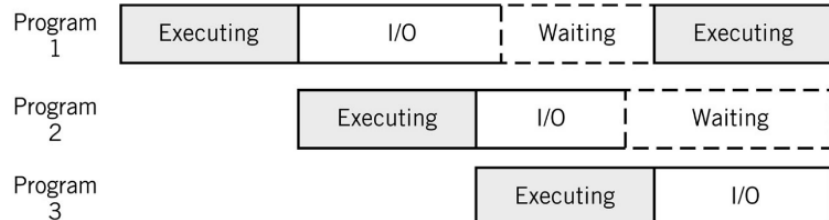
	<ul style="list-style-type: none"> ○ Infrequently used programs, software tools, and commands. ● Bootstrap program ● Diskless workstations or thin clients = programs, including the OS, are located on another computer on the network. <p>Simplified Diagram of Operating System Services:</p>  <pre> graph TD User[User] <--> App[Application program] User <--> CI[Command interface] App <--> CI CI <--> FS[File services] CI <--> NS[Network services] FS <--> NS FS <--> CS[Core services] NS <--> CS CS <--> IOS[I/O services] IOS <--> CH[Computer hardware] IOS <--> NH[Network hardware] subgraph OS [Operating system] CI FS NS CS IOS end </pre>
OS Activities	<p>OS Degree of Activity</p> <ul style="list-style-type: none"> ● Interactive (Conversational Systems) ● Batch processing <ul style="list-style-type: none"> ○ Little to no user interactions. <ul style="list-style-type: none"> ■ Users submit programs or jobs. ● Event driven - interrupts or service requests. <p>Hardware and the OS</p> <ul style="list-style-type: none"> ● A standard operating system that works on different hardware: <ul style="list-style-type: none"> ○ Provide program and file portability. ○ Enable user efficiency through recognizable interfaces. ○ Is implemented through a systems programming language like C or C++. <p>Single Job Processing</p> <ul style="list-style-type: none"> ● Only one program is loaded into memory and executed. <ul style="list-style-type: none"> ○ Example: MS-DOS ● Memory resident components: <ul style="list-style-type: none"> ○ Command interface shell. ○ I/O routines, including BIOS. ○ File management system. ● User program in control. <ul style="list-style-type: none"> ○ When the program is finished control is transferred back to the command interpreter.

- The user can stop the program execution via a keyboard interrupt.
- Disadvantages:
 - Lack of security
 - Programs can overwrite the resident OS.
 - Programs can write directly to I/O devices.
 - Minimum memory management and no scheduling.
 - Often idle CPU awaiting the completion of I/O operations.

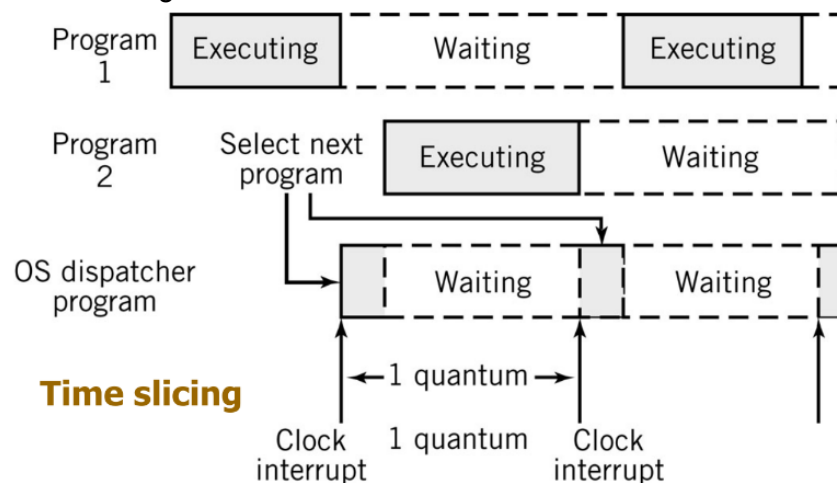
Concurrent Operations

- Multitasking (multiprogramming) vs. multiprocessing with multiple CPUs
 - Multitasking
 - Another program would use the CPU while one program is waiting for the I/O operations.
 - Time-slicing - Rapidly switched CPU between programs.
 - Dispatching = the process of selecting which program to run.
- Concurrent processing vs. simultaneous processing

Sharing the CPU during I/O Breaks:



Time-sharing the CPU:



<p>Services and Facilities</p>	<p>Services and Facilities</p> <ul style="list-style-type: none"> • Command processor • File management system • I/O control system • Process control management and interprocess communication • Memory management • Scheduling system • Secondary storage management • System protection management • Network management, communication support, and communication interfaces • System Administration <p>User Interface and Command Execution Services</p> <ul style="list-style-type: none"> • Types of user interfaces: <ul style="list-style-type: none"> ◦ CLI - Command Line Interface ◦ GUI - Graphical User Interface ◦ Menu environment • Shell = user interface and command processor that interacts with the kernel. <ul style="list-style-type: none"> ◦ Example: UNIX - C, Bourne and Korn shells • Command Languages: <ul style="list-style-type: none"> ◦ IBM Mainframes – JCL. ◦ MS Windows – BAT files, Windows Scripting Host. ◦ UNIX – shell scripts. <p>File Management</p> <ul style="list-style-type: none"> • File = logical unit of storage. • Basic file management system: <ul style="list-style-type: none"> ◦ Directory structures for each I/O device. ◦ Tools to copy and move files. ◦ Information about each file and the accessing tools. ◦ Security mechanisms to protect files and control access. • Additional file management features: <ul style="list-style-type: none"> ◦ Backup, emergency retrieval and recovery. ◦ File compression. ◦ Transparent network file access. ◦ Auditing. <p>I/O Services and Process Control Management</p> <ul style="list-style-type: none"> • I/O services: <ul style="list-style-type: none"> ◦ Startup configuration. ◦ Device drives that implement interrupts and provide other techniques for handling I/O. ◦ Plug and play: hot swapping, hot plugging. • Process control management:
--------------------------------	---

- Process = executing programs.
- Thread = individually executable part of a process.
- Interprocess messaging services - example: a temporary pipe between UNIX or DOS.

Memory Management

- Keep track of memory
- Maintain queues of waiting programs.
- Allocate memory to the following program.
- Deallocate a program's memory space upon program completion.

Scheduling

- High-level scheduling = placed in queue based on level of priority and eventually executed.
- Dispatching = actual selection of processes that will be executed.
 - Preemptive - use clock interrupts.
 - Non-preemptive - voluntary control transfer by the program.
- Context switching = transfer control to the process that is being dispatched.
 - Preemptive v.s. non-preemptive.
- Processing requirements - CPU v.s. I/O bound.

Secondary Storage and Security

- Secondary storage management:
 - Optimize completion of I/O tasks for efficient disk usage.
 - Combines both hardware and software.
- Security and protection services:
 - Protect OS from users.
 - Protect users from other users.
 - Prevent unauthorized entry to the system.
 - Prevent unauthorized system use by authorized users.

Network and Communication Services

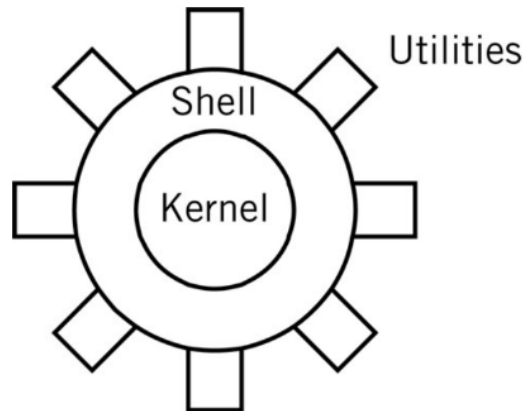
- TCP-IP protocol suite:
 - Locate and connect to other computers.
 - Access files, I/O devices, and programs from remote systems.
 - Support distributed processing.
- Network applications - examples: email, remote login, web services, streaming multimedia, voice over IP telephony, VPN
- Interface between communication software and OS I/O control system that provides network access.

	<p>System Administration Support</p> <ul style="list-style-type: none">• System configuration and setting group configuration policies.• Modification on user list.• Modification on user privileges.• System security.• Files systems management.• Network administration.• Backups.• Software installations and upgrades.• OS installations (system generation), patches, and upgrades.• System tuning and optimization.
--	---

Kernel

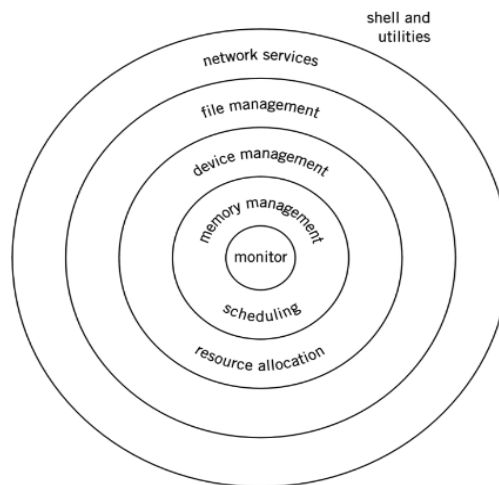
Monolithic Kernel

- Drawback: Low stability and integrity
- Examples: UNIX, Windows NT



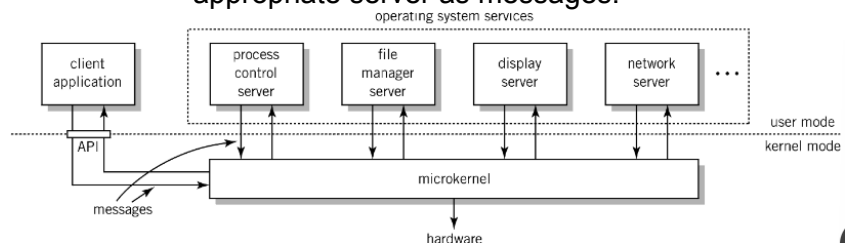
Hierarchical

- Requests pass through intermediate layers.
- Examples: Multics, Data General



Microkernel

- Provide minimum essential functionalities.
- Provide client-server systems on the same system.
 - Requested services are passed onto the appropriate server as messages.



Mach OS Kernel

- Implement the Microkernel structure, but includes services of:
 - Message passing
 - Interrupt processing
 - Virtual memory management
 - Scheduling
 - Basic set of I/O drivers
- Examples: Macintosh OS X, IBM AIX on RS/6000.

Types of Operating Systems

- Single user, single tasking.
- Single user, multitasking.
- Multi-user, multitasking.
- Distributed systems = process power distribution among computers in a cluster or network,
- Network servers
- Real-time systems
- Embedded systems

The Process of Bootstrapping

- Executions stored in ROM begin, with a bootstrap loader (mini-loader - IPL).
- The OS program is looked at in a fixed location.
- The OS is loaded into RAM.
- The control is transferred to the starting location of the OS.
- User programs are loaded and executed by the loader program in the OS.

Bootstrapping

- Cold v.s. warm boot (no system reset)

