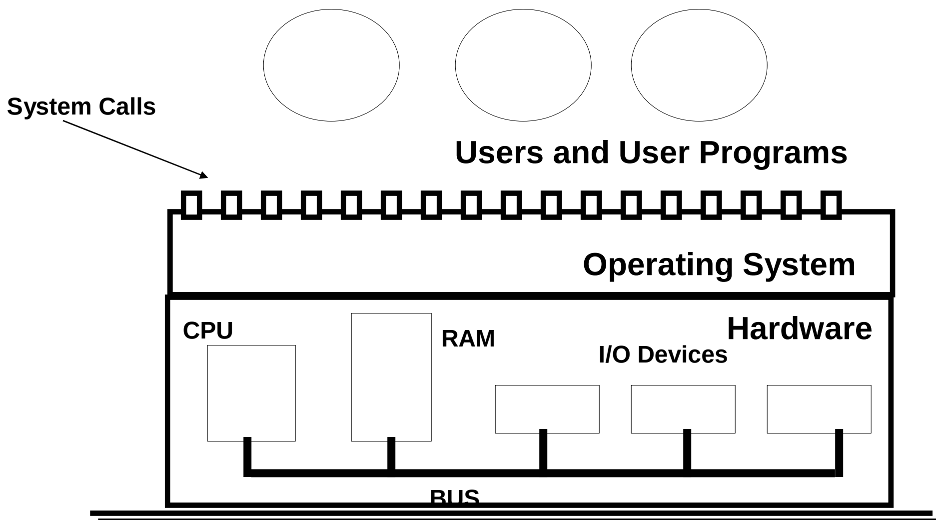
The computer system



Processes and Programs.

Single Programming.

OS allows (can handle)

- ♦ only one process executing
- ♦ only one process somewhere between start and finish

Multi-Programming.

OS allows

- ♦ only one process to be executing
- ♦ multiple processes to be between start and finish

Processes and Programs CONT.

Multi-Processing.

OS (and h/w) allows

- ♦ **more than one process to be executing**
- ♦ **more than one process to be between start and finish.**

The trend is towards multi-programming and multi-processing systems.

Why Multi-Programming?

Generally multi-programming is used to interleave I/O and computation to increase efficiency.

CPUs capable of 16 Million instructions per second.

Disks capable of accessing one block of information in 0.002 seconds. so how many instructions in this time???

An Exercise.

Program A	Program B	Program C
100 instructions	read 1 sector	1000 instructions
write 1 sector	100 instructions	write 1 sector
100 instructions	write 1 sector	

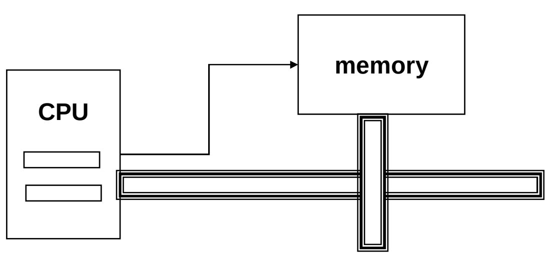
Read/Write 1 sector = 0.0020 seconds
execute 100 instructions = 0.001 seconds

With both single and multi-programming

- **How long does it take to execute all 3 programs?**
- **How much time does the CPU do nothing**

Storage Structure

- **Main Memory:**
  - ♦ **Only large storage media that CPU can access directly.**
  - ♦ **Access to register data is quite fast. But access to memory data is very slow.**
  - ♦ **Even then all programs cannot be stored in mainmemory**
    - » **Cost**
    - » **Volatility.**



Storage Structure

- **Main Memory:**
  - ♦ **Only large storage media that CPU can access directly**
- **Secondary Storage:**
  - ♦ **extension of the main memory that provides nonvolatile storage capacity**
- **Magnetic Disks:**
  - ♦ **rigid metal or glass platters covered with magnetic recording material**
  - ♦ **Disk surface is logically divided into *tracks*, which are subdivided into *sectors***
  - ♦ **The disk controller determines the logical interaction between the devided and computer**

Storage Hierarchy

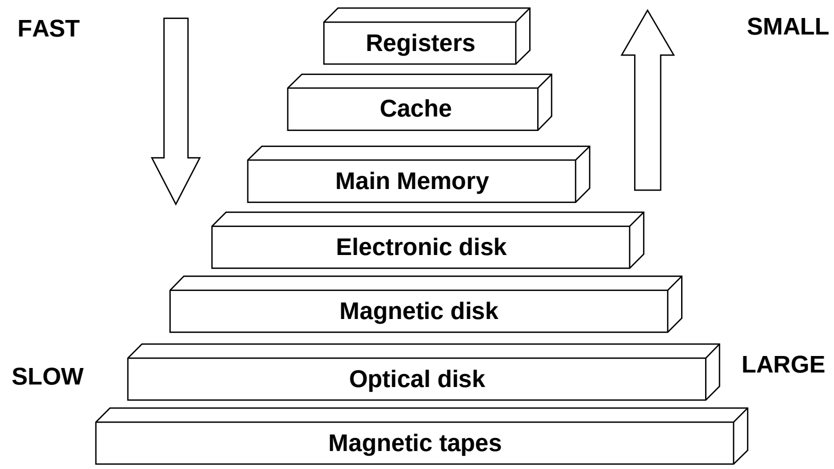
Storage systems organized in hierarchy:

- ♦ **Speed**
- ♦ **cost**
- ♦ **volatility**

Caching:

- ♦ **copying information into faster storage system; main memory can be viewed as a fast cache for secondary storage**

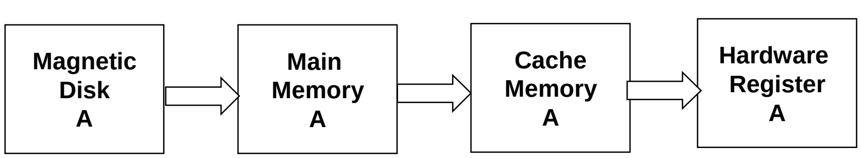
Storage-Device Hierarchy



Coherency and Consistency

One data can appear in several places. If it is modified in one place, all copies have to be updated.

The problem becomes complex with multiprocessing.



Hardware Protection Mechanisms

Sharing requires protection of resources (CPU, IO, memory) from each other (OS - user process, or user process- user process).

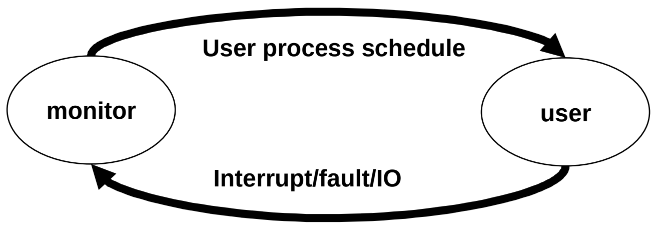
- ♦**Dual-Mode operation**
- ♦**I/O Protection**
- ♦**Memory Protection**
- ♦**CPU Protection**

Dual-Mode Operation

- **Sharing system resources requires operating system to ensure that an incorrect program cannot cause other programs to execute incorrectly.**
- **Provide hardware support to differentiate between at least two modes of operations.**
  - ♦ **User mode: execution done on behalf of a user.**
  - ♦ **Monitor mode (supervisor mode or system mode): execution done on behalf of operating system**

Dual-Mode Operation (cont.)

- **Mode bit added to computer hardware to indicate the correct mode: monitor (0) or user (1).**
- **When interrupt or fault occurs hardware switches to monitor mode.**
- **Privileged instructions can be issued only in monitor mode.**



I/O Protection

- **OS must ensure that I/O devices are not monopolized, two user process do not attempt to write simultaneously.**
- **All I/O instructions are privileged instructions**

General-System Architecture

Given that I/O instructions are privileged, how does the user program perform I/O?

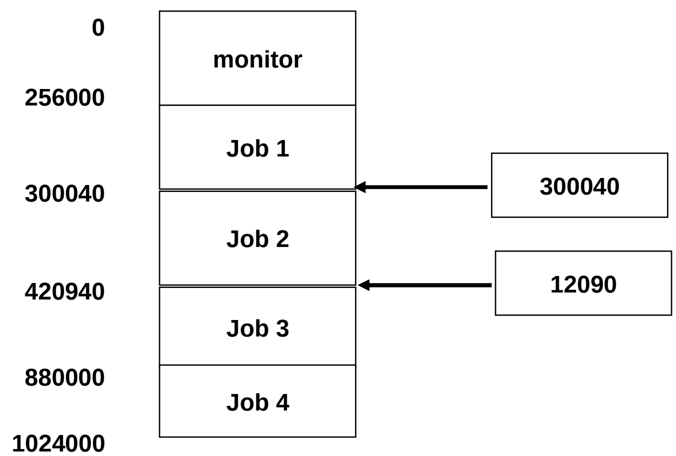
System call- the method used by a process to request action by the operating system

- ♦ Usually takes the form of a trap to a specific location in the interrupt vector
- ♦ Control passes through the interrupt vector to a service routine in the OS, and the mode bit is set to monitor mode.
- ♦ The monitor verifies that the parameters are correct and legal, executes the request, and returns control to the instruction following system call.

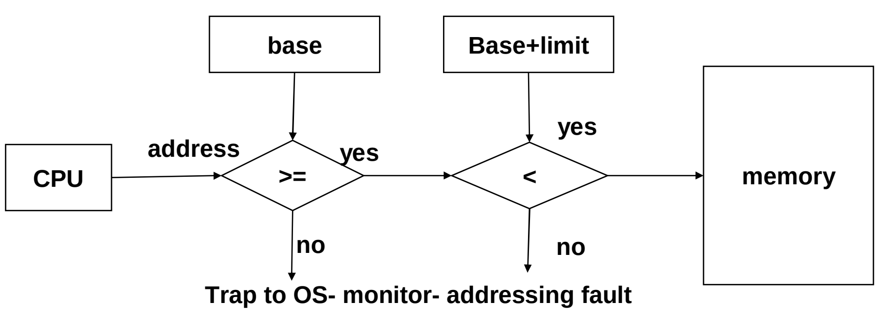
Memory Protection

- OS must ensure that a user program could never gain control of the computer in monitor mode
  - ♦ (i.e. a user program that, as part of its execution, stores a new address in the interrupt vector, or the memory location where OS is residing)
- In order to have memory protection add two registers that determine the legal range of addresses a user program may access:
  - ♦ base register
  - ♦ limit register
- Memory outside the defined range is protected.

Example of Memory Protection



Protection Hardware



- When executing in monitor mode, the OS has unrestricted access to both monitor and users' memory.
- The load instruction for the base and limit registers are privileged instructions.

CPU Protection

- **Timer- interrupts computer after specified period to ensure operating system maintains control**
  - ♦ **Timer is decremented every clock tick**
  - ♦ **when timer reaches the value of 0, an interrupt occurs**
- **Timer commonly used to implement time sharing**
- **Timer also used to compute the current time**
- **Load-timer is a privileged instruction**