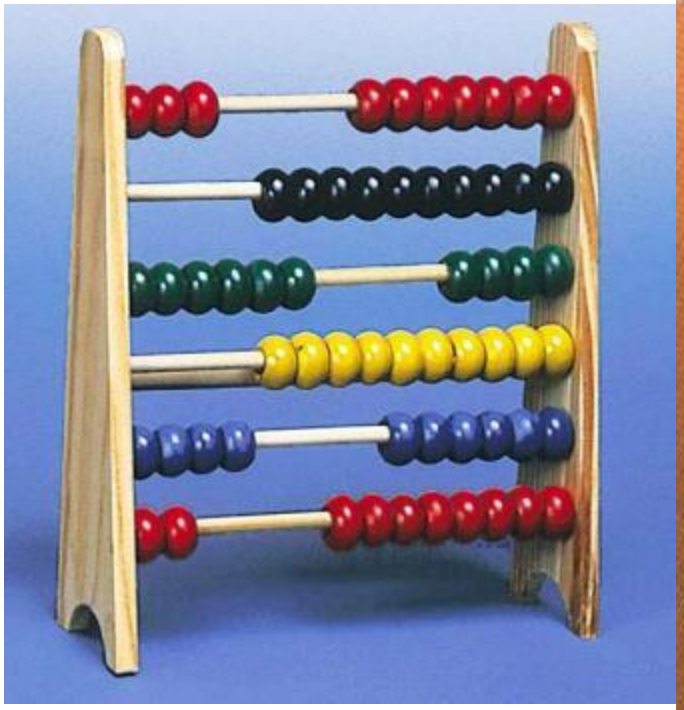
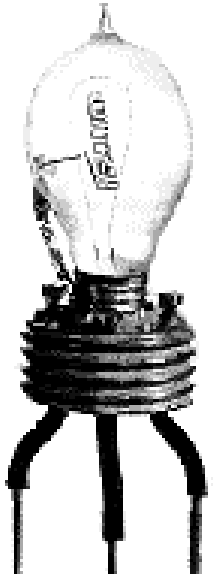
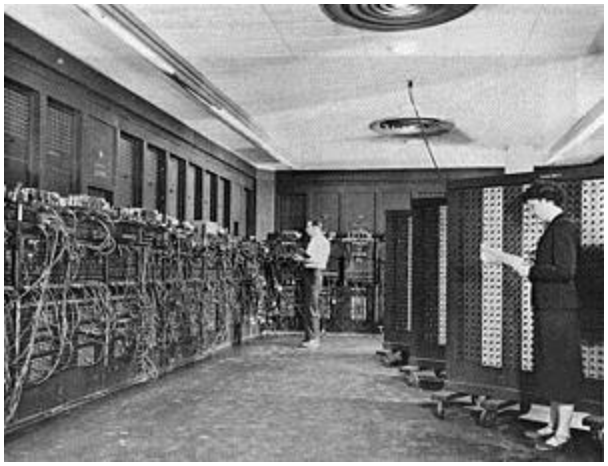
When we start?



Early Systems



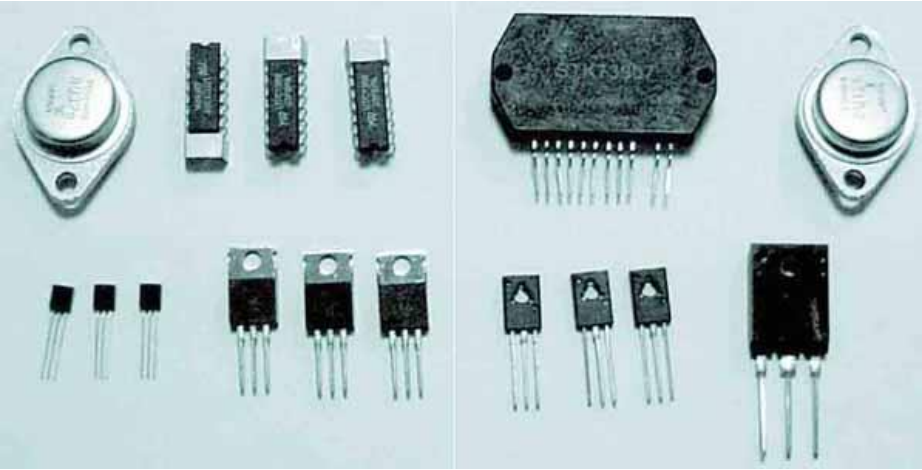
vacuum tube from the early 1900's



- Early Systems
  - 1945 – 1955
  - Bare machines – vacuum tubes and plug-boards
  - No operating system
  - No protection
  - ENIAC – Electronic Numerical Integrator And Computer

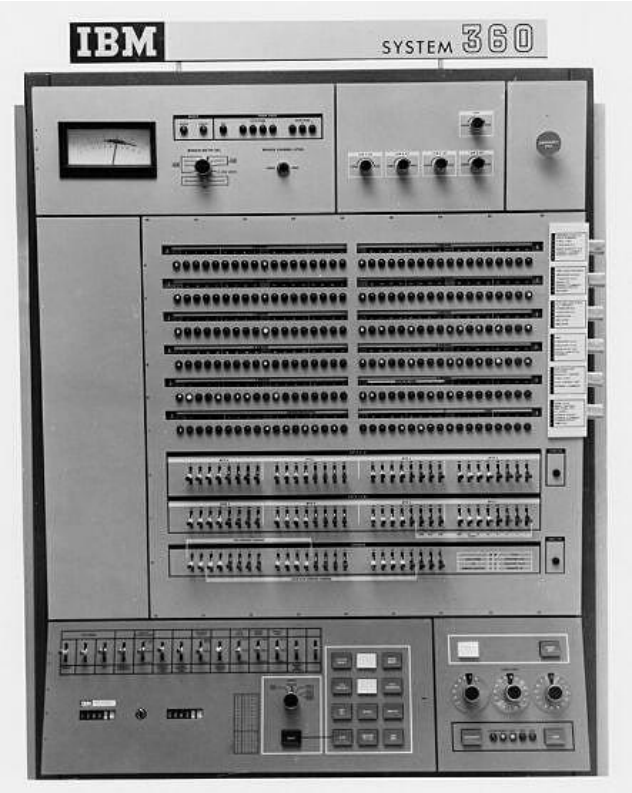
Second Generation

- Second Generation Systems
  - 1956 – 1965
  - Transistors and batch systems
  - I/O channel
  - Interrupts / exceptions
  - Minimal protection
  - Libraries / JCL



Third Generation

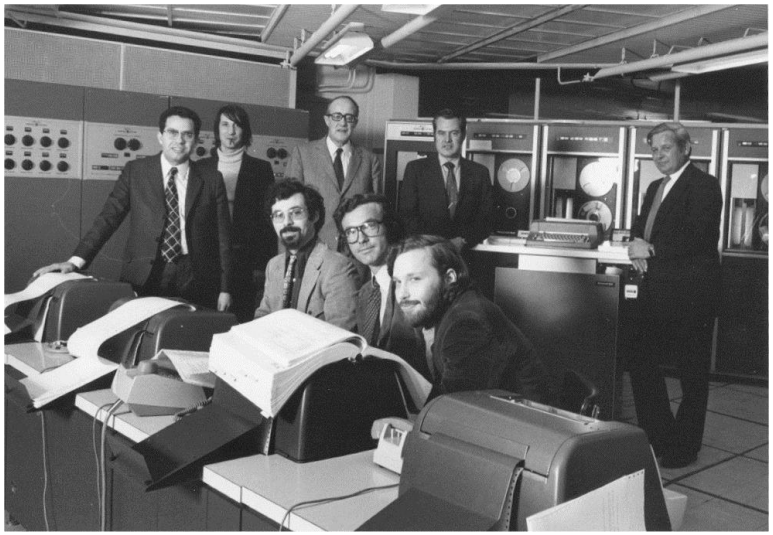
- 1965 – 1980
- ICs and Multiprogramming
- System 360 and S/370 family of computers
- Spooling (simultaneous peripheral operation on-line)
- Time sharing
- On-line storage for
  - System programs
  - User programs and data
  - Program libraries
- Virtual memory
- Multiprocessor configurations
  - Local multiprocessing – All CPUs connected through a local bus
  - Distributed multiprocessing – CPUs connected through a network-like connection



MULTICS

MULTICS – Multiplexed Information and Computing Service

- Design started in 1965 and completed in 1972
- Collaborative effort between General Electric, Bell Telephone Labs, and Project MAC of MIT
- Aimed at providing
  - simultaneous computer access to large community of users
  - ample computation power and data storage
  - easy data sharing between users, if desired



Fourth Generation …

- Personal computers and workstations
- MS-DOS , Linux, Windows NT
- Massively parallel systems
  - Pipelining
  - Array processing / SIMD
  - General multiprocessing / MIMD
  - Symmetric multiprocessing / SMP
- Computer networks (communication aspect) – network operating systems
- Distributed computing – distributed operating systems
- Cluster computing, grid computing, etc.
- Mobile computing (mobile operating systems)

Modern classification based on use

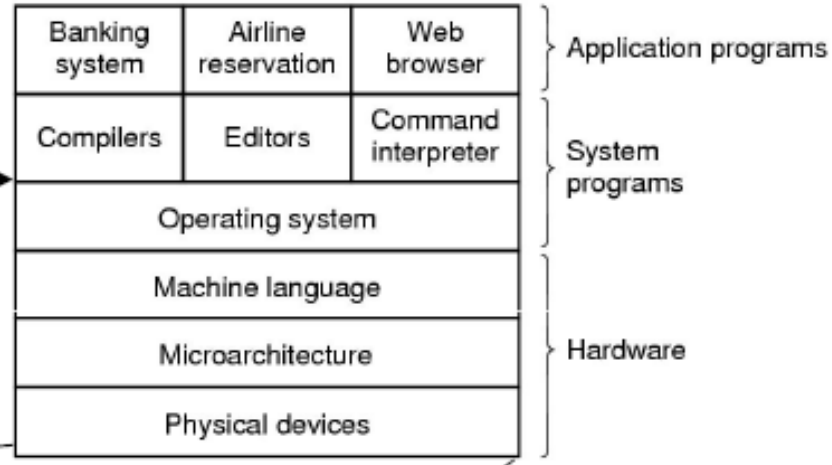
- Low-end consumer desktop
  - Assumes no intelligence on part of the user
  - Focuses on hardware and software compatibility
- High-end desktop
  - User is smart but may not be very computer savvy
  - Must provide good performance for applications, such as CAD/CAM
  - Provide hardware and software compatibility
- Real-time systems
  - User is professional programmer
  - Focus on performance, defined response time, easy extendable hardware support, and programming control
- Embedded systems
  - Closed box system to do specific functions well
  - No expansion slots (standard machine has multiple standard interfaces)
  - Limited applications
  - Turn it on and use it (standard host enables all components, scans every interface, and is ready to run multiple apps)

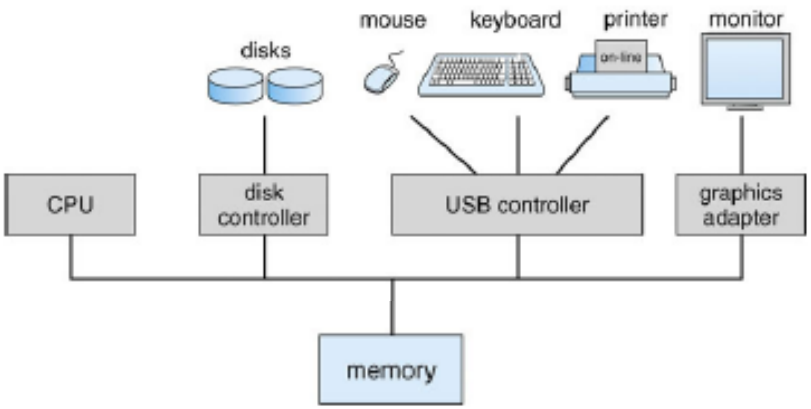
Where is Operating System?

A computer system consists of

- hardware
- system programs
- application programs

separation between kernel and user mode



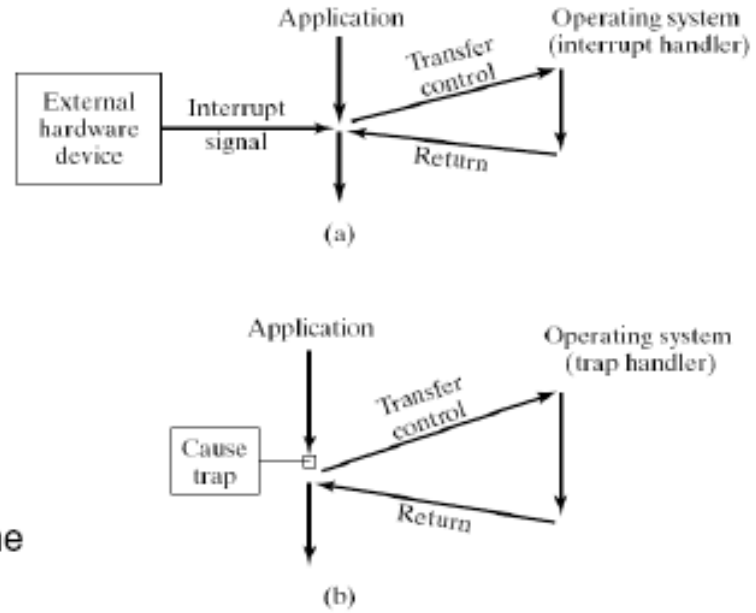


ABSTRACTION
- Simple, easier to use interface (machine-independent)
- Hiding of unnecessary details

Operating System modes

Two *modes*, supported by the processor

- supervisor (kernel, privileged) and user mode
- some processors support a series of modes for increasing privileges
  - Intel: 4 *rings*



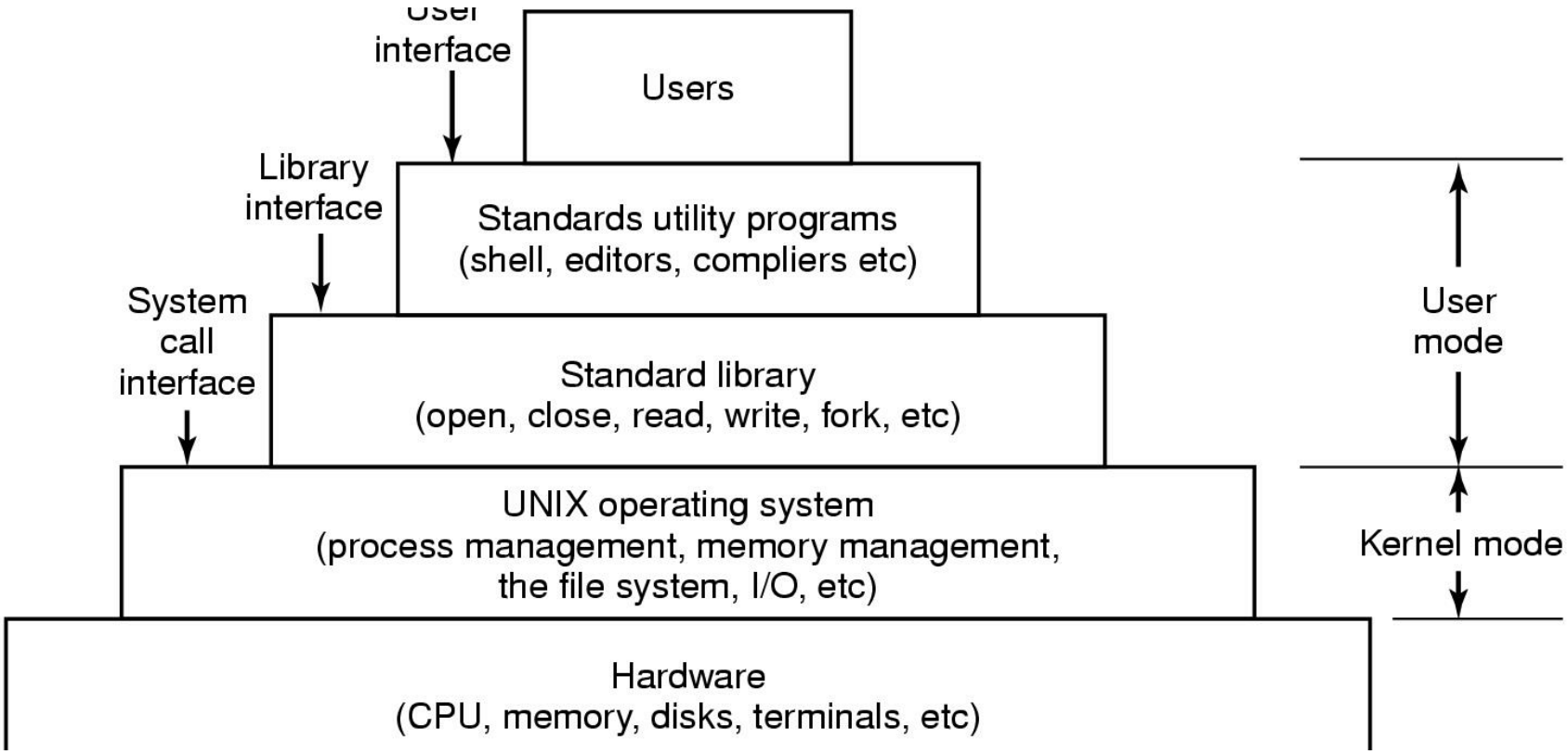
Transfer of control to kernel
(i.e., transfer to code executed in kernel mode)

- interrupt: external source
  - e.g. clock tick, i/o completion
- trap: internal (software) source
  - system call (explicit SuperVisor Call or SVC), in line with control flow of program
  - generated software error (e.g. floating point error)

Using these modes, *guarantees* can be given regarding protection

Example OS: UNIX

- developed in 1969 by a group of AT&T employees at Bell Labs.
- was first developed in assembly, but by 1973 almost entirely recoded in C.



Two different views of an OS

- Extended machine view
  - Virtual machine that is easier to understand and program
  - Tool to make programmer's job easy
- Resource manager view
  - Tool to facilitate efficient operation of computer system
  - Provides services to users; processor, memory, I/O, system bus

Elements of an OS

- **Processor/CPU**
  - Controls the operations of computer and performs data processing functions
  - Exchanges data with memory using memory address register and memory buffer register
  - Exchanges data with I/O devices using I/O address register and I/O buffer register
- **Main memory/Primary memory**
  - Stores data and programs
  - Typically volatile
- **I/O modules**
  - Moves data between a computer and external environment
  - Communicates with a variety of devices including secondary memory (disks), communications equipment, and

terminals

- **System bus**
  - Provides for communications among processors, main memory, and I/O modules

OS Main Components

- Process management
  - process creation; deletion; suspension
  - process synchronization; communication
- Main-memory management
  - Manage used parts and their current users
  - Select processes to load
  - Allocate memory to running processes
- Secondary storage management
  - Free-space management
  - Storage allocation

OS Main Components

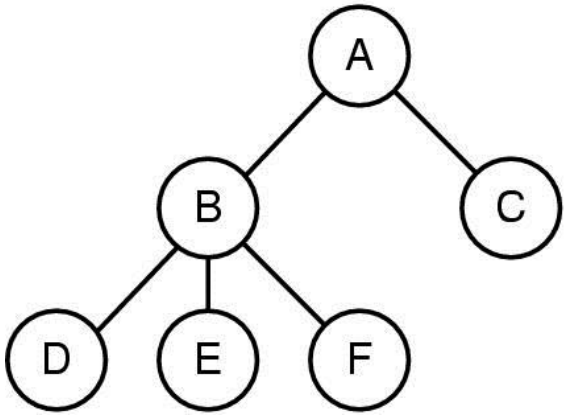
- File system management
  - File + directory creation; deletion
  - File manipulation primitives
  - Mapping files onto secondary storage
- I/o system management
  - general device-driver interface
  - Drivers for specific hardware devices
- Protection system
  - Distinguish between authorized and unauthorized usage
  - Provide means of enforcement

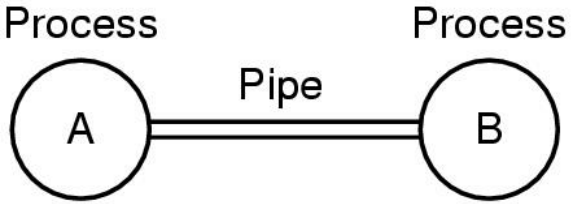
OS Concepts - Program

- **Program**
  - Collection of instructions and data kept in ordinary file on disk
  - Executable program, complete code output by linker/loader, with input from libraries
    - Generated from source program and object code
  - The file is marked as executable in the i-node
  - File contents are arranged according to rules established by the kernel

OS Concepts - Process

- **Processes**
  - Created by kernel as an environment in which a program executes
  - May be stopped and later restarted by the OS
  - Process may acquire resources (more memory, open files) not present in the program
  - Process Tree: Child and parent processes
  - Communication between processes through messages



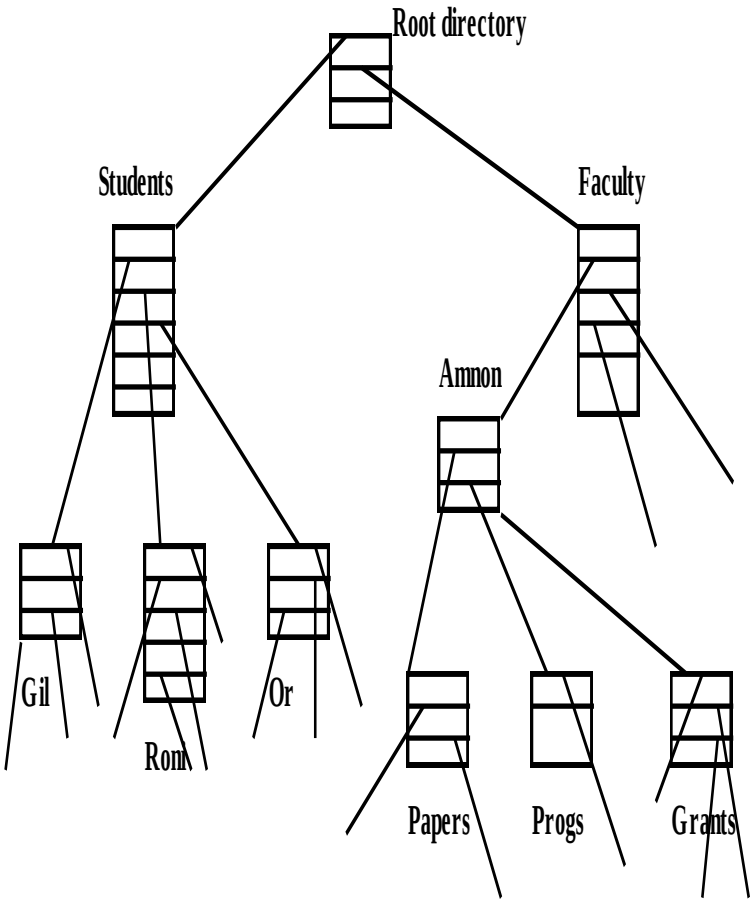


OS Concepts - Threads

- **Threads**
  - Stream of instruction execution; running instance of a process' code
  - A dispatchable unit of work to provide intra-process concurrency
  - A process may have multiple threads of execution in parallel, each thread executing sequentially
  - Threads may execute in time-sharing manner on a single CPU, on multiple cores concurrently, or on multiple CPUs

OS Concepts - Files

- **Files**
  - Services of file management system to hide disk/tape specifics
  - System calls for file management
  - Directory to group files together
  - Organized as a hierarchical tree
  - Root directory
  - Path name
  - Path name separator
  - Working directory
  - Protection of files (9-bit code in Unix – rwx bits)
  - File descriptor or handle – small integer to identify a file in subsequent operations, error code to indicate access denied

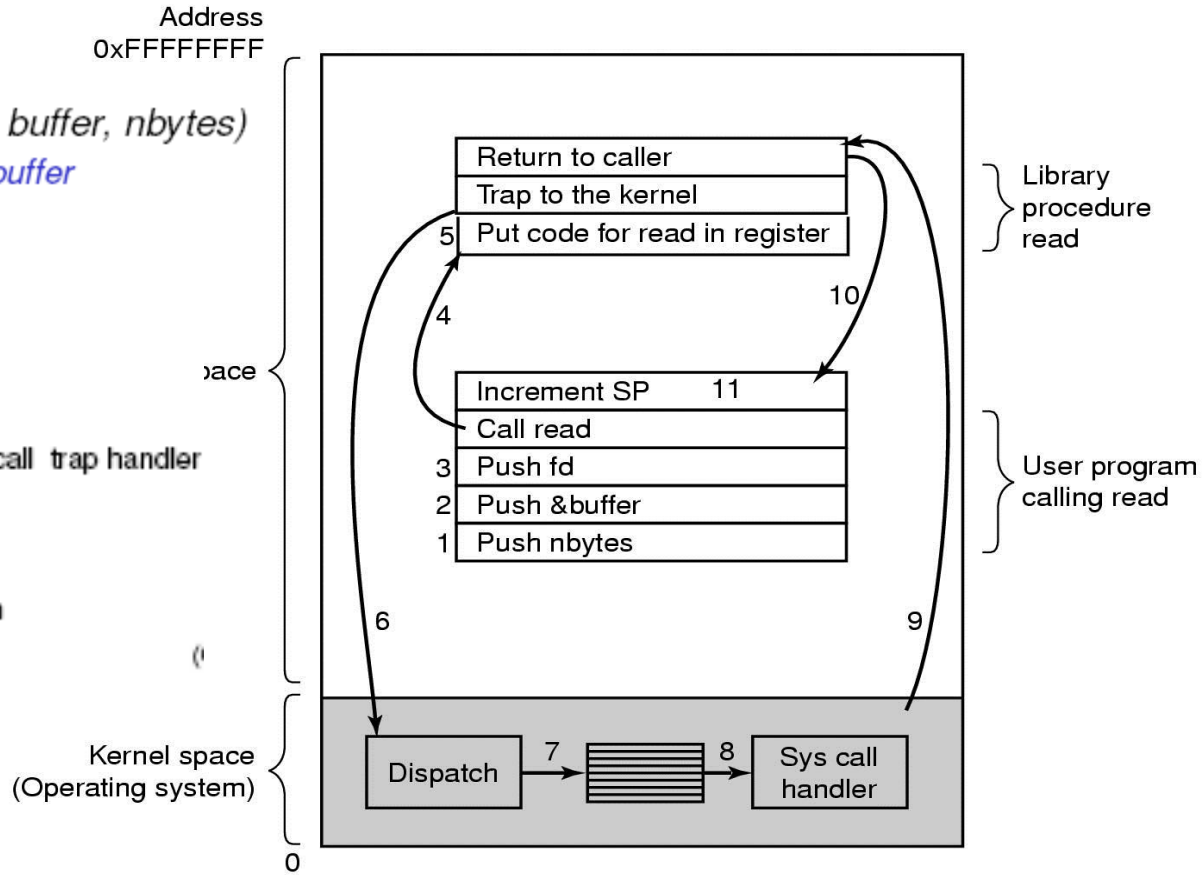


OS Concepts – System Calls

- **System calls**
  - Interface between user program and operating system
  - Set of extended instructions provided by the operating system
  - Applied to various software objects like processes and files
  - Invoked by user programs to communicate with the kernel and request services
  - Broadly divided into six classes:
    - filesystem (open), process (fork), scheduling (priocntl), IPC (semop), socket/networking (bind), and miscellaneous (time)

A System Call

Execute *C* statement: *status = read(fd, buffer, nbytes)*

- read *nbytes* bytes from *fd*, storing it in *buffer*

1-3: pushing parameters
4: call library function
5: put code for *read* in reg.
6: trap (Linux@x86: int 0x80): switch mode and call trap handler
7: handler calls read function handler
8: handler performs read actions (a.o. store data at address). Here *suspension of the calling process* may occur if data needs to come from an io device.
9-11: control back to caller