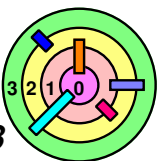


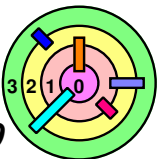
1960s OS Issues

- ➡ **Multiprogramming (i.e., running things "in parallel" with one CPU)**
- ➡ **Time sharing (i.e., support interactive users)**
- ➡ **Software complexity**
- ➡ **Security**



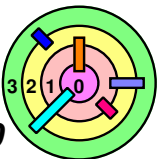
2010s OS Issues

- ➡ **Multiprogramming (i.e., running things "in parallel" with one CPU)**
 - not just one computer, but server farms
- ➡ **Time sharing (i.e., support interactive users)**
 - voice, video, sound, etc.
- ➡ **Software complexity**
 - a bigger problem than could be imagined in the 1960s
- ➡ **Security**
 - ditto



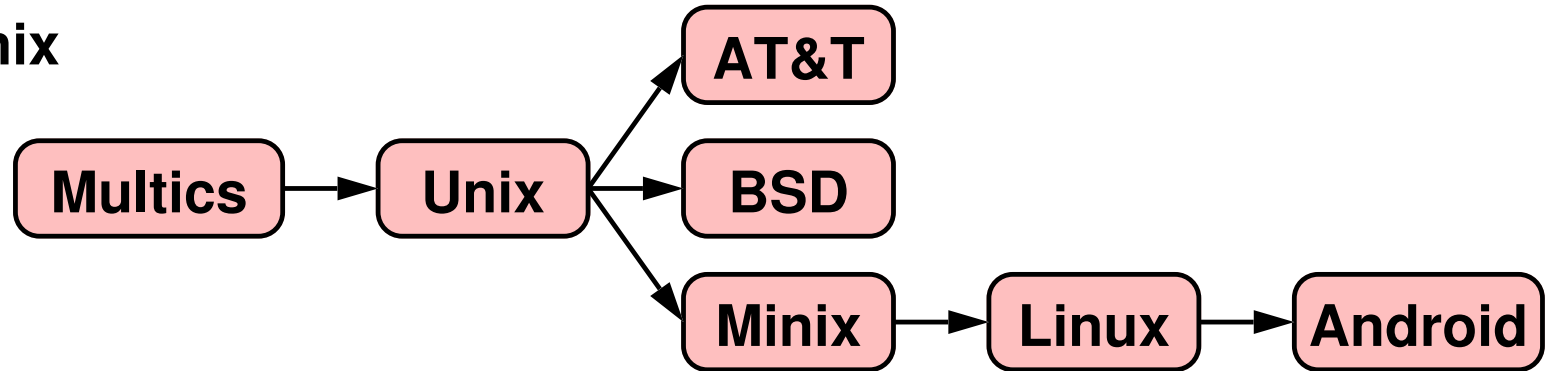
1.2 A Brief History of Operating Systems

- ➡ **The 1950's: The Birth of the Concept**
- ➡ **The 1980's: The Modern OS Takes Form**
- ➡ **Minicomputers & Unix**
- ➡ **The Personal Computer**

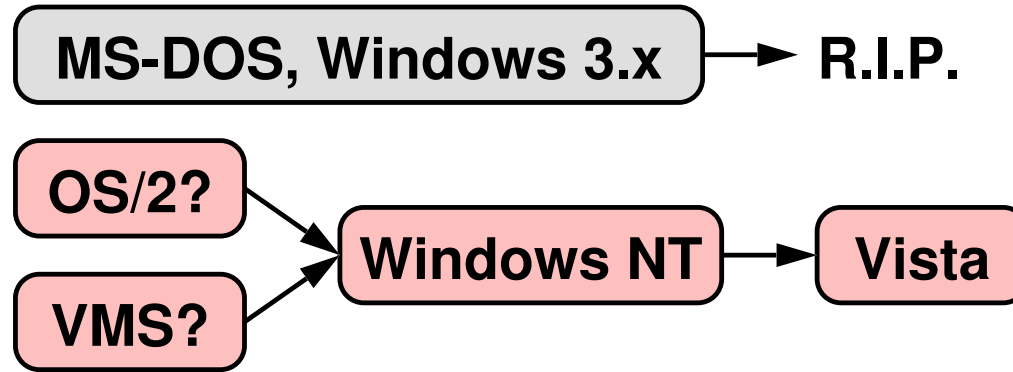


Where Do Things Evolve From?

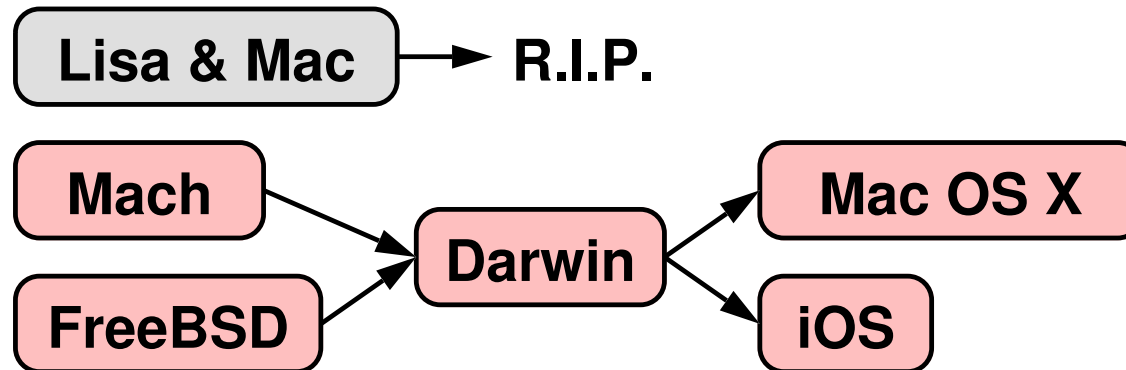
➡ Linux & Unix



➡ Microsoft

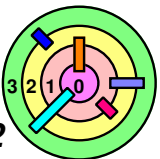


➡ Apple

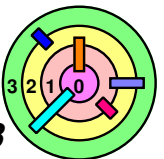


History of C

- ➡ **Early 1960s: CPL (Combined Programming Language)**
 - developed at Cambridge University and University of London
- ➡ **1966: BCPL (Basic CPL): simplified CPL**
 - intended for systems programming
- ➡ **1969: B: simplified BCPL (stripped down so its compiler would run on minicomputer)**
 - used to implement earliest Unix
- ➡ **Early 1970s: C: expanded from B**
 - motivation: they wanted to play "Space Travel" on minicomputer
 - used to implement all subsequent Unix OSes
- ➡ **Unix has been written in C ever since**



Extra Slides



In the Beginning ...



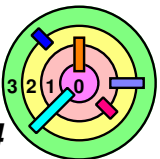
There was hardware

- processor
- storage
- card reader
- tape drive
- drum



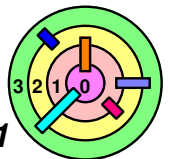
And not much else

- no operating system
- no libraries
- no compilers
- very little software in the beginning



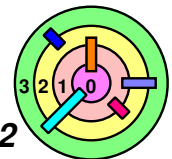
1.3 A Simple OS

- ➡ *OS Structure*
- ➡ **Processes, Address Spaces, & Threads**
- ➡ **Managing Processes**
- ➡ **Loading Program Into Processes**
- ➡ **Files**



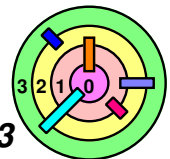
A Simple OS

- ➡ The main focus of this class is on how to *build an OS*
 - since this is an intro class, we will focus on the fundamentals
 - occasionally, we will talk about the more advanced topics
 - this is not a "tech" class
- ➡ Sixth-Edition Unix
 - source license available to universities in 1975 from Bell Labs
 - had major influence on modern OSes
 - Solaris
 - Linux
 - MacOS X
 - Windows
- ➡ Fits into 64KB of momory
 - single executable, completely stored in a *single file*
 - *loaded* into memory as the OS boots
 - *monolithic OS*



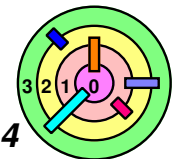
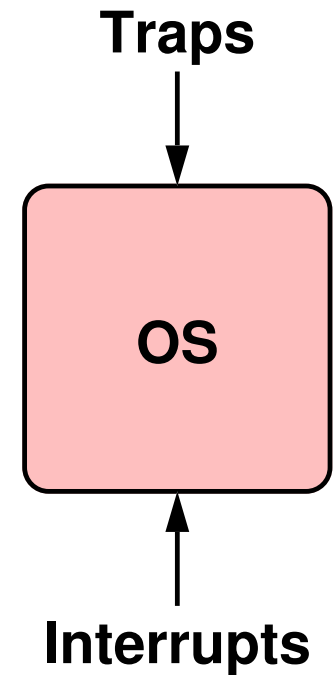
Hardware Support - User vs. Privileged Modes

- ➡ **Processor modes:** part of the processor state (recall from your computer organization/architecture class regarding "processor")
 - most computers have at least two modes of execution
 - **user mode:** fewest privileges
 - **privileged mode:** most privileges
 - ◆ the only code that runs in this mode is part of the OS
- ➡ **For Sixth-Edition Unix**
 - the whole OS run in the privileged mode
 - everything else is an application and run in the user mode
- ➡ **For other systems**
 - major subsystems providing OS functionality may run in the user mode
- ➡ **We use the word "*kernel*" to mean the portion of the OS that runs in privileged mode**
 - sometimes, a subset of this



A Simple OS Structure

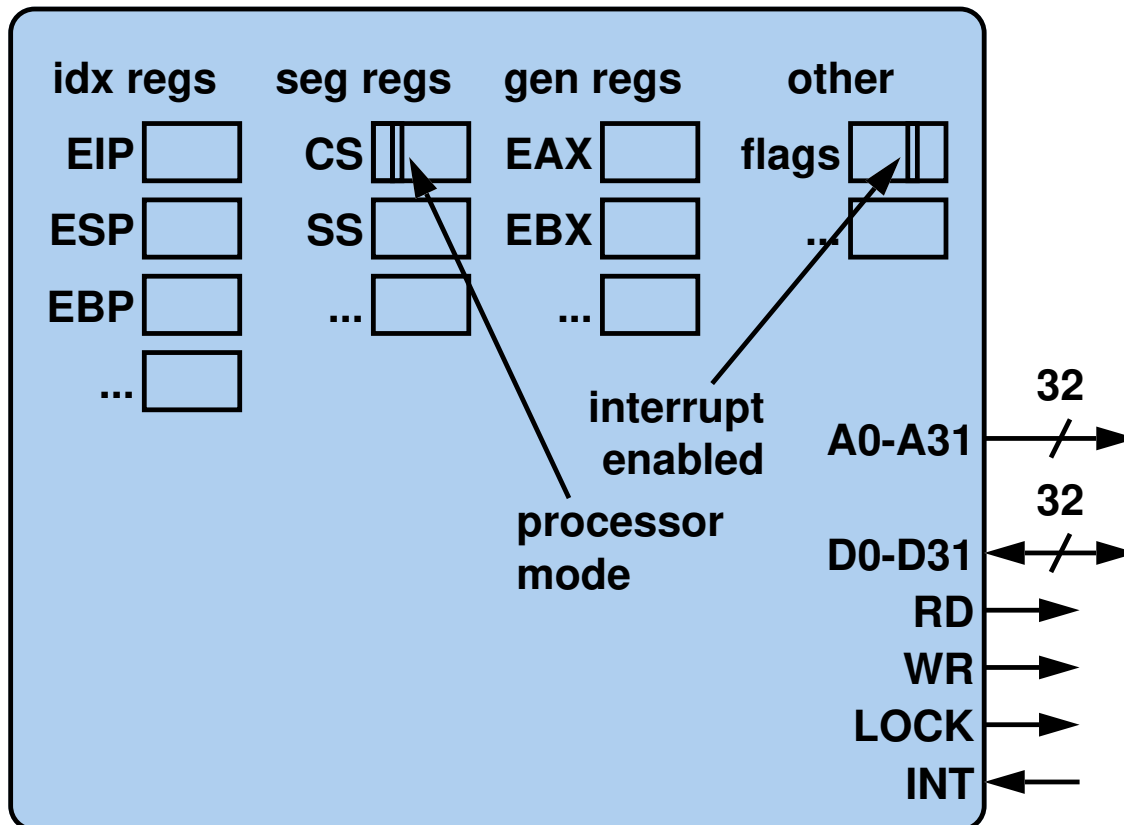
- ➡ Application programs call upon the OS via *traps*
- ➡ External devices call upon the OS via *interrupts*
 - ▬ I/O completion interrupt
 - executes *interrupt service routine*



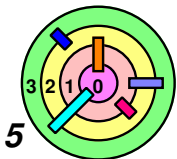
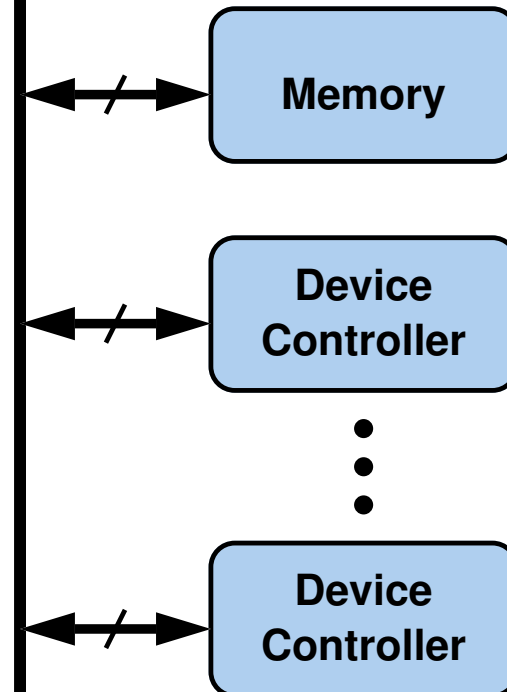
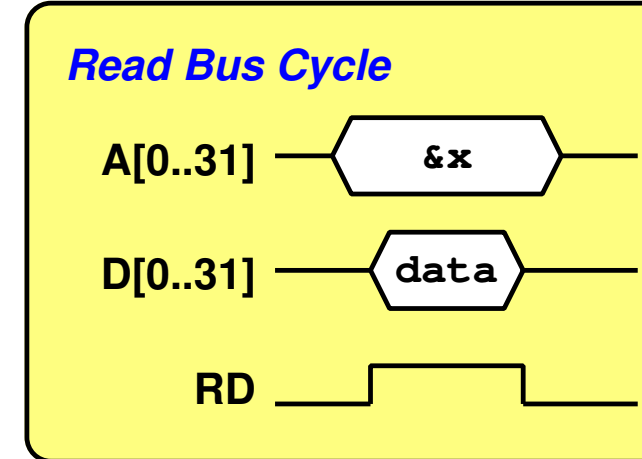
A Simple OS Structure

- ➡ Review of "Computer Organization"
- bus architecture

x86 Processor



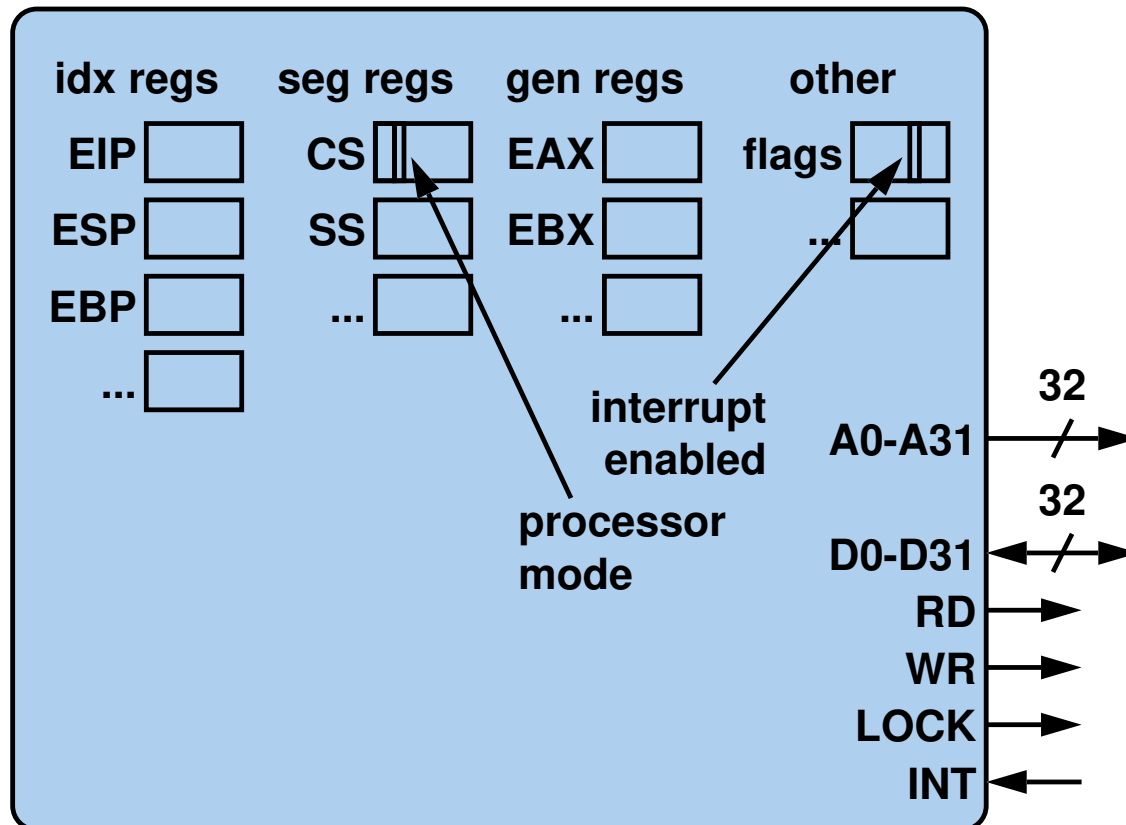
Bus



A Simple OS Structure

- ➡ Review of "Computer Organization"
- bus architecture

x86 Processor



Bus

