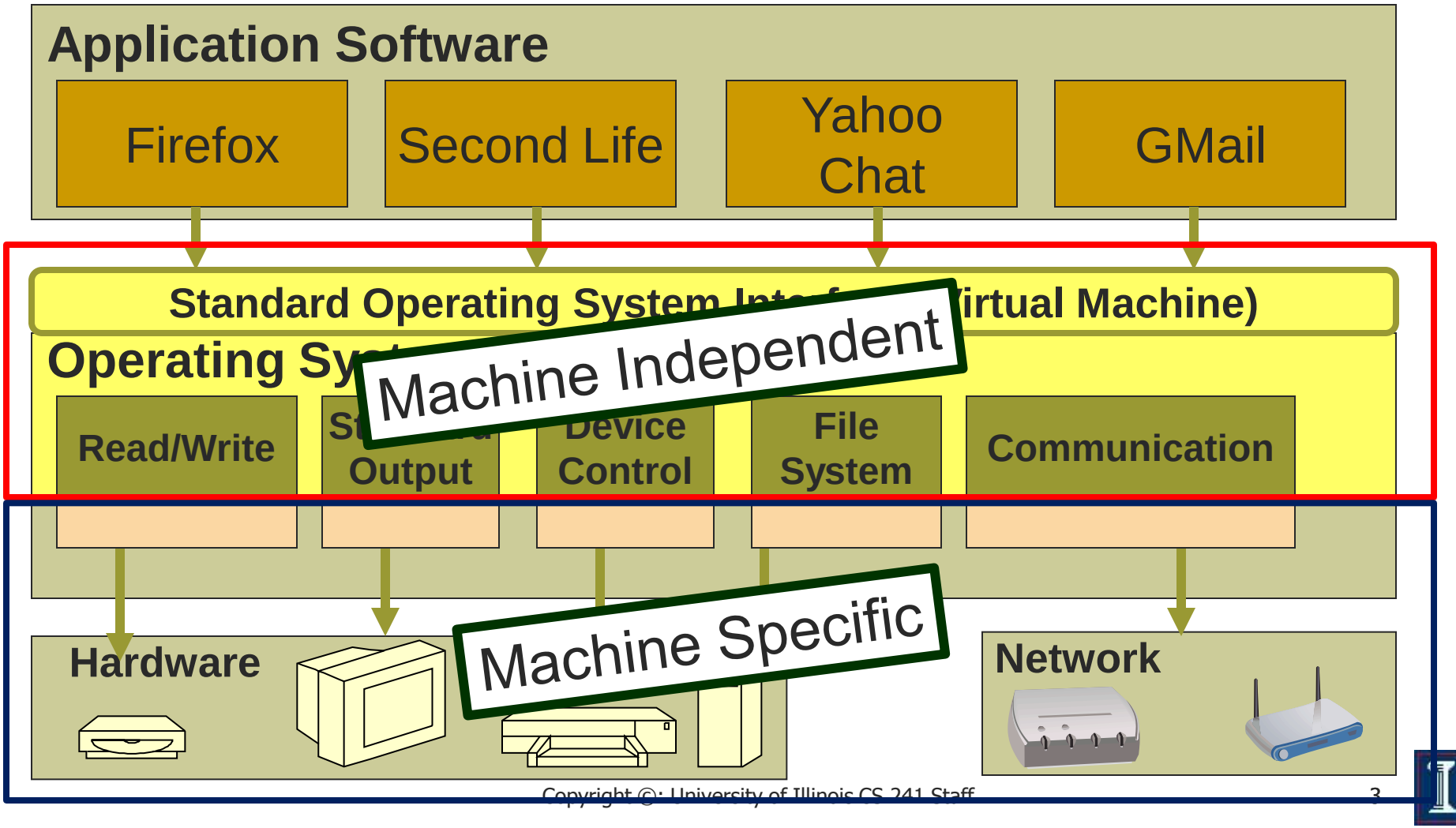
Operating Systems Orientation

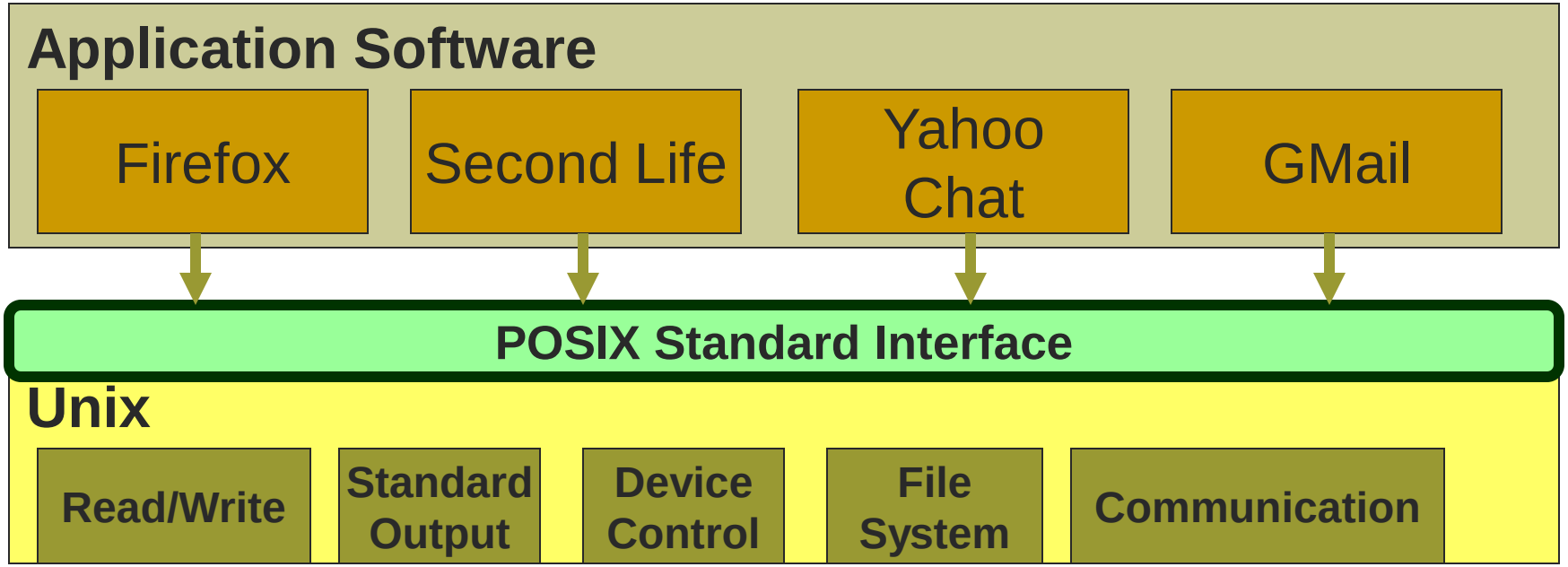
Objectives

- Explain the main purpose of operating systems and describe milestones of OS evolution
- Explain fundamental machine concepts
  - Instruction processing
  - Memory hierarchy
  - Interrupts
  - I/O
- Explain fundamental OS concepts
  - System calls
  - Processes
  - Synchronization
  - Files
- Explain the POSIX standard (UNIX specification)

OS Structure



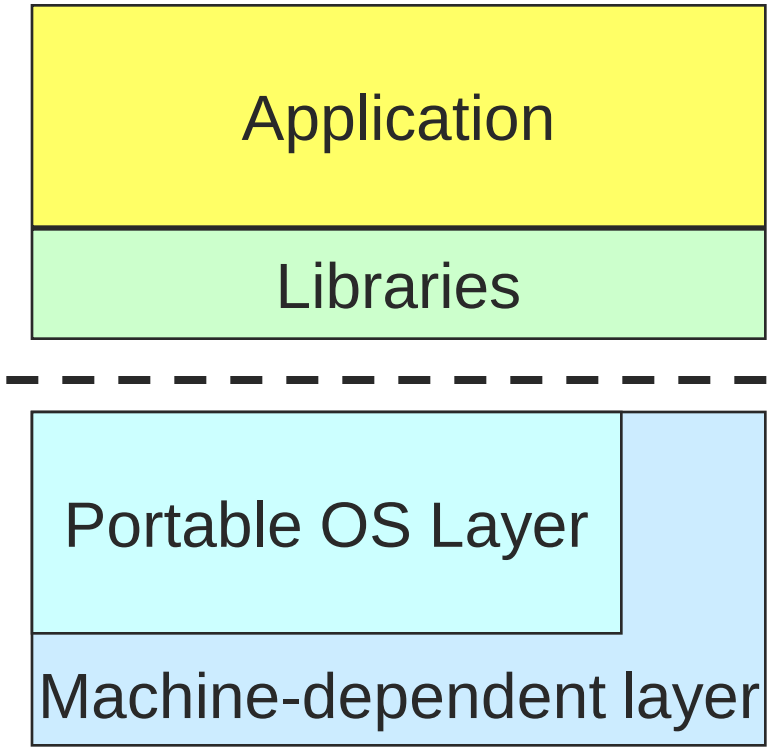
POSIX
The UNIX Interface Standard



What is an Operating System?

- It is an *extended machine*
  - Hides the messy details that must be performed
  - Presents user with a virtual machine interface, easier to use
- It is a *resource manager*
  - Each program gets time with the resource
  - Each program gets space on the resource

A Peek into Unix

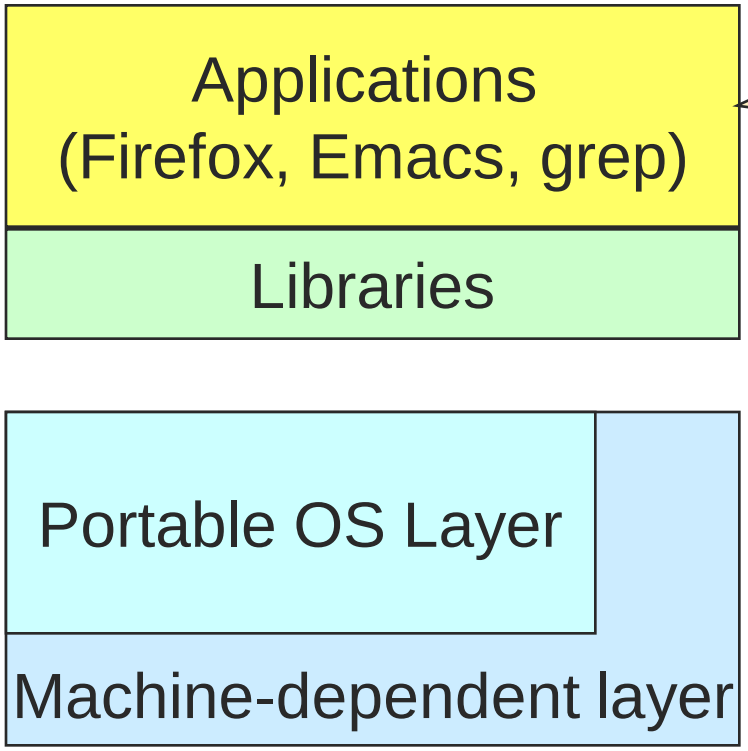


User space/level

Kernel space/level

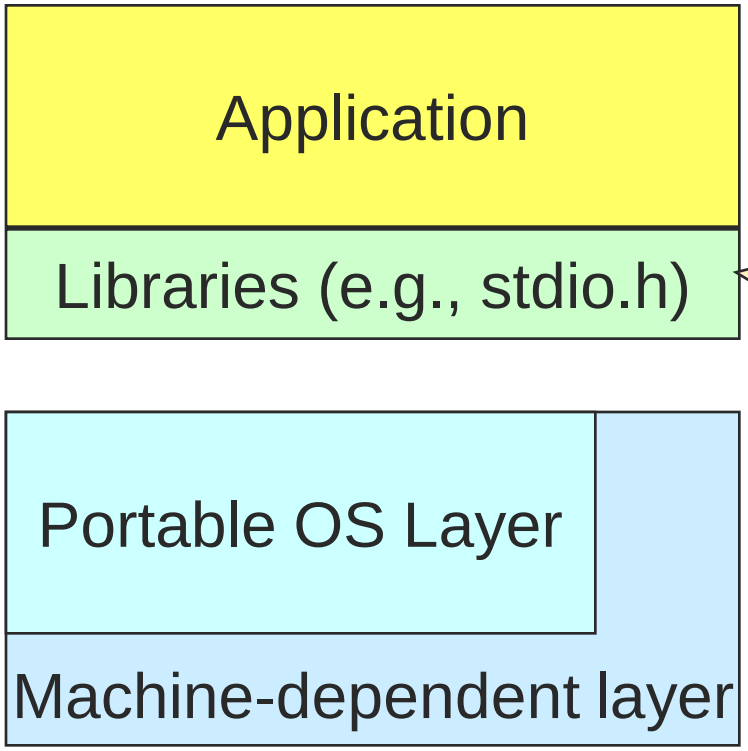
- **User/kernel modes are supported by hardware**
- **Some systems do not have clear user-kernel boundary**

Application



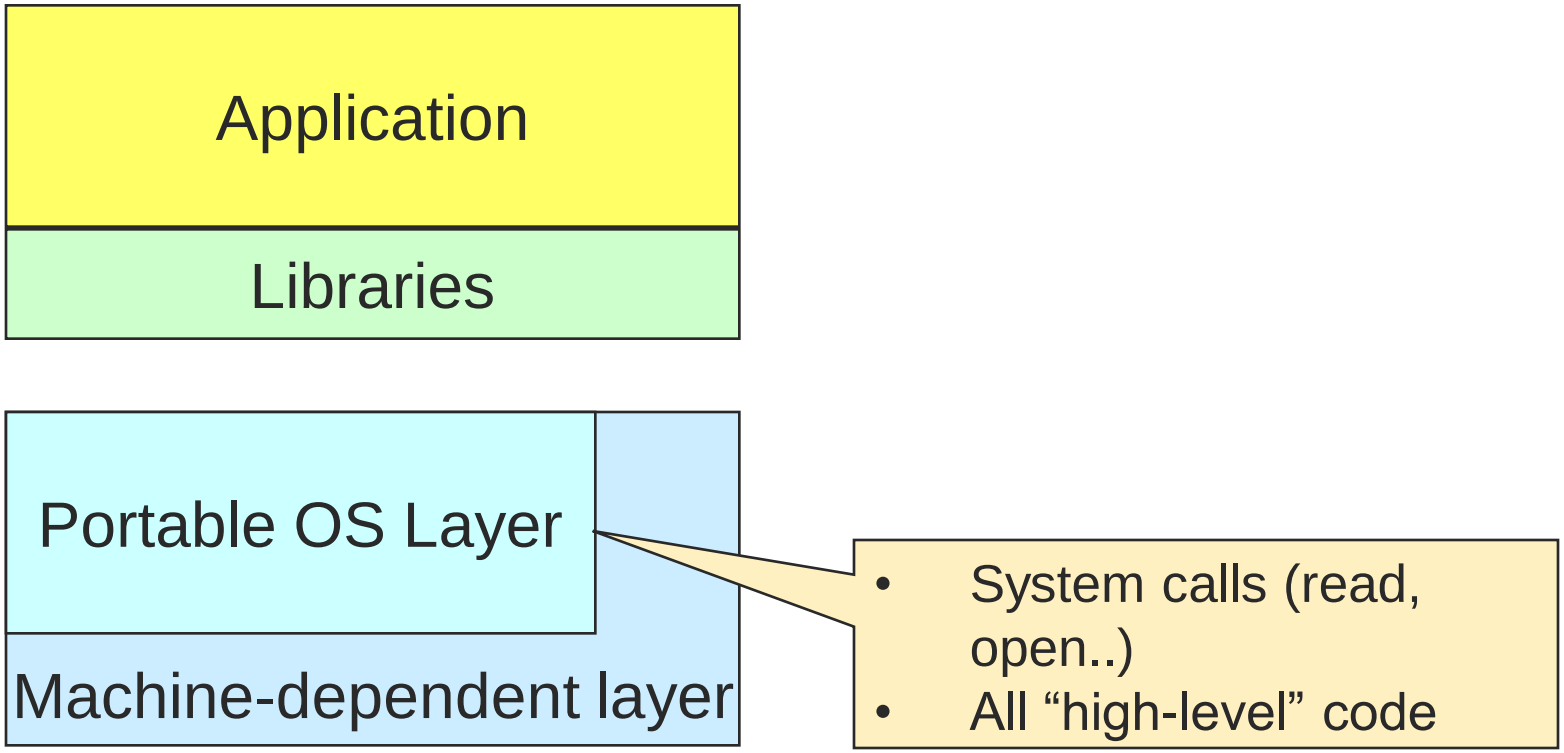
- Written by programmer
- Compiled by programmer
- Uses function calls

Unix: Libraries

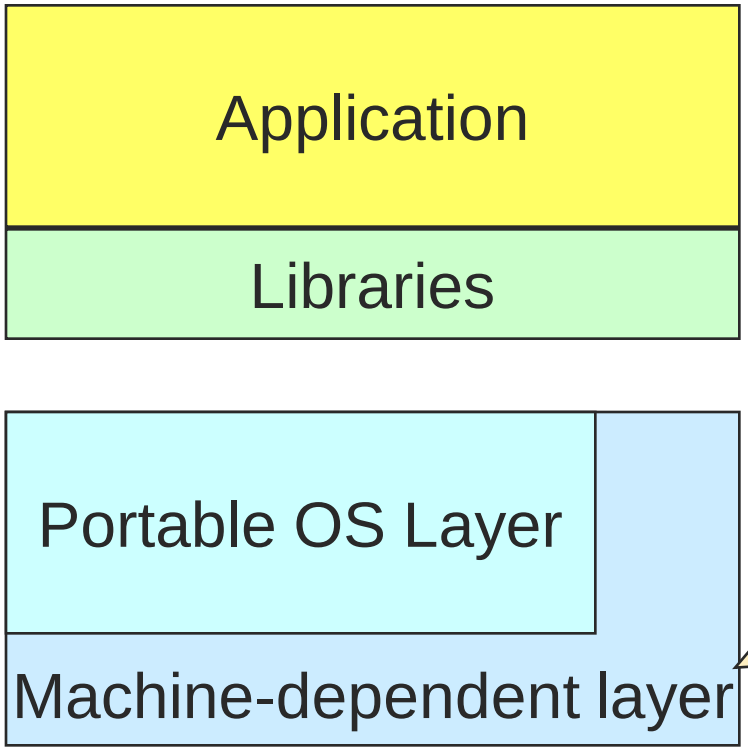


- Provided pre-compiled
- Defined in headers
- Input to linker (compiler)
- Invoked like functions
- May be “resolved” when program is loaded

Typical Unix OS Structure



Typical Unix OS Structure

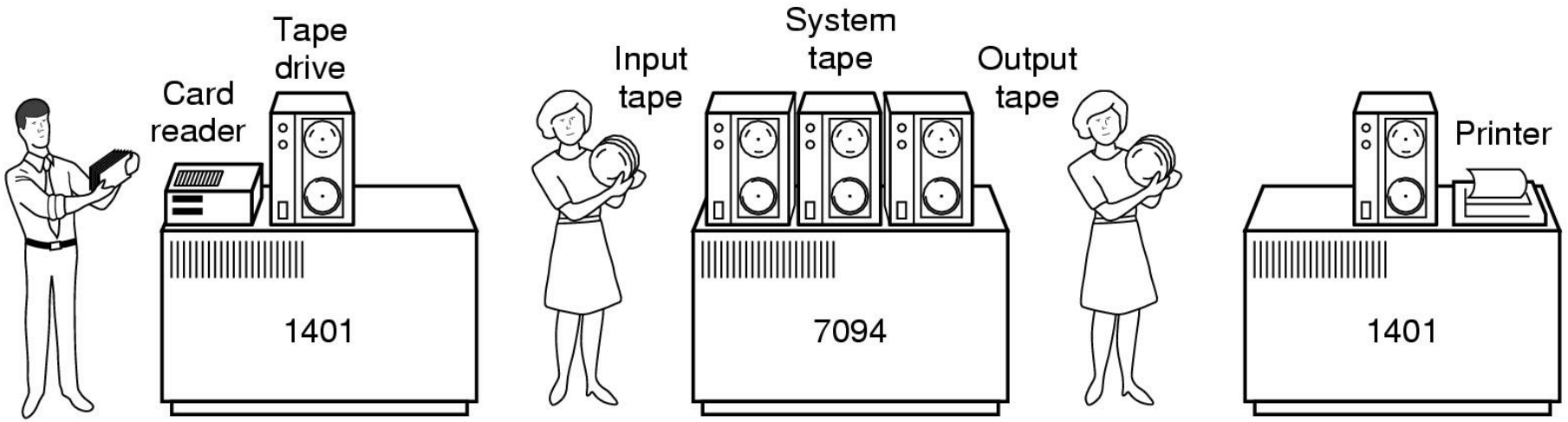


- Bootstrap
- System initialization
- Interrupt and exception
- I/O device driver
- Memory management
- Kernel/user mode switching
- Processor management

History of Computer Generations

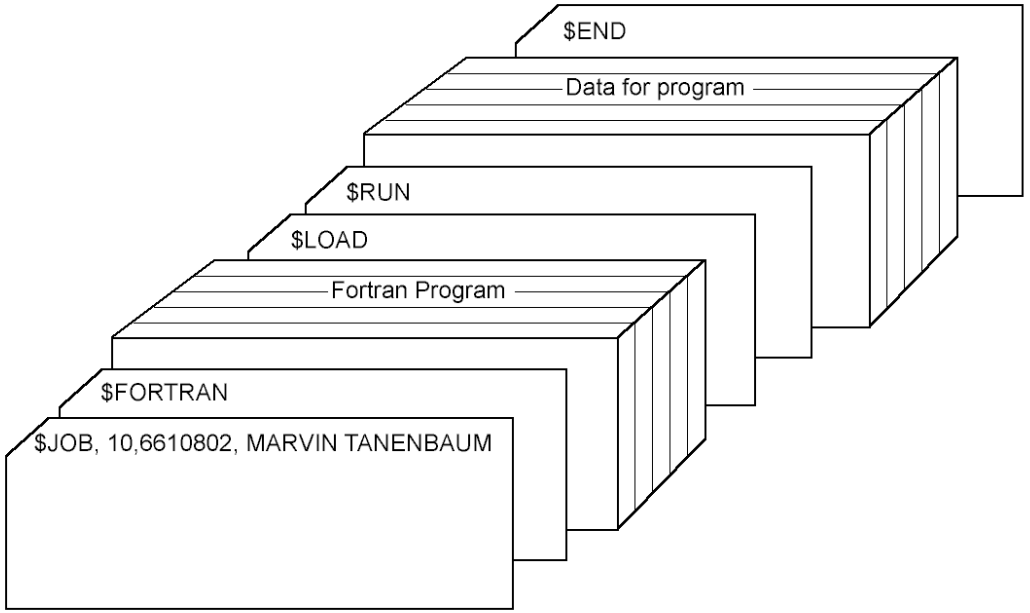
- Pre-computing generation 1792 - 1871
  - Babbage's "Analytical Engine", first programming language
- First generation 1945 – 1955
  - Vacuum tubes, plug boards
- Second generation 1955 - 1965
  - Transistors, batch systems, mainframes
- Third generation 1965 – 1980
  - ICs and multiprogramming
- Fourth generation 1980 – present
  - Personal computers

History of Operating Systems



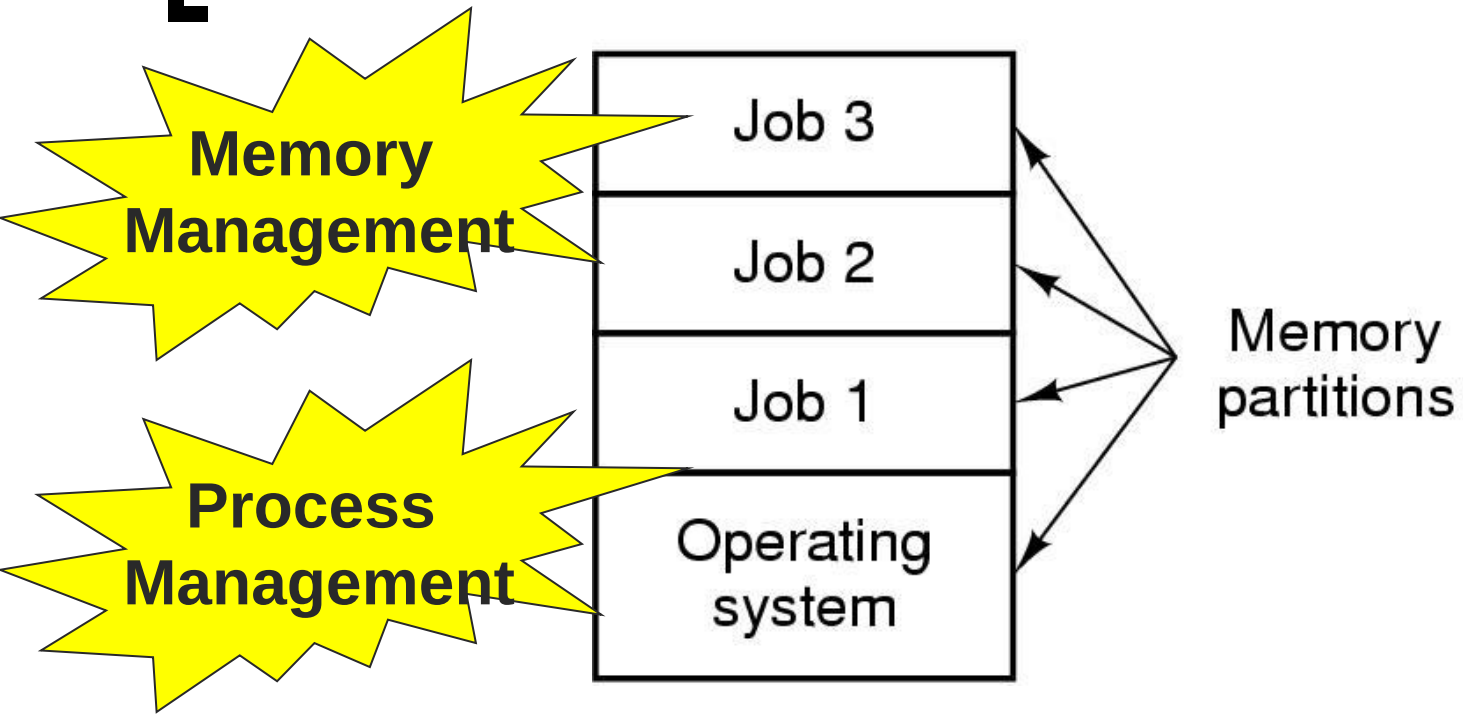
- Early systems
  - bring cards to 1401
  - read cards to tape
  - put tape on 7094 which does computing
  - put tape on 1401 which prints output

History of Operating Systems



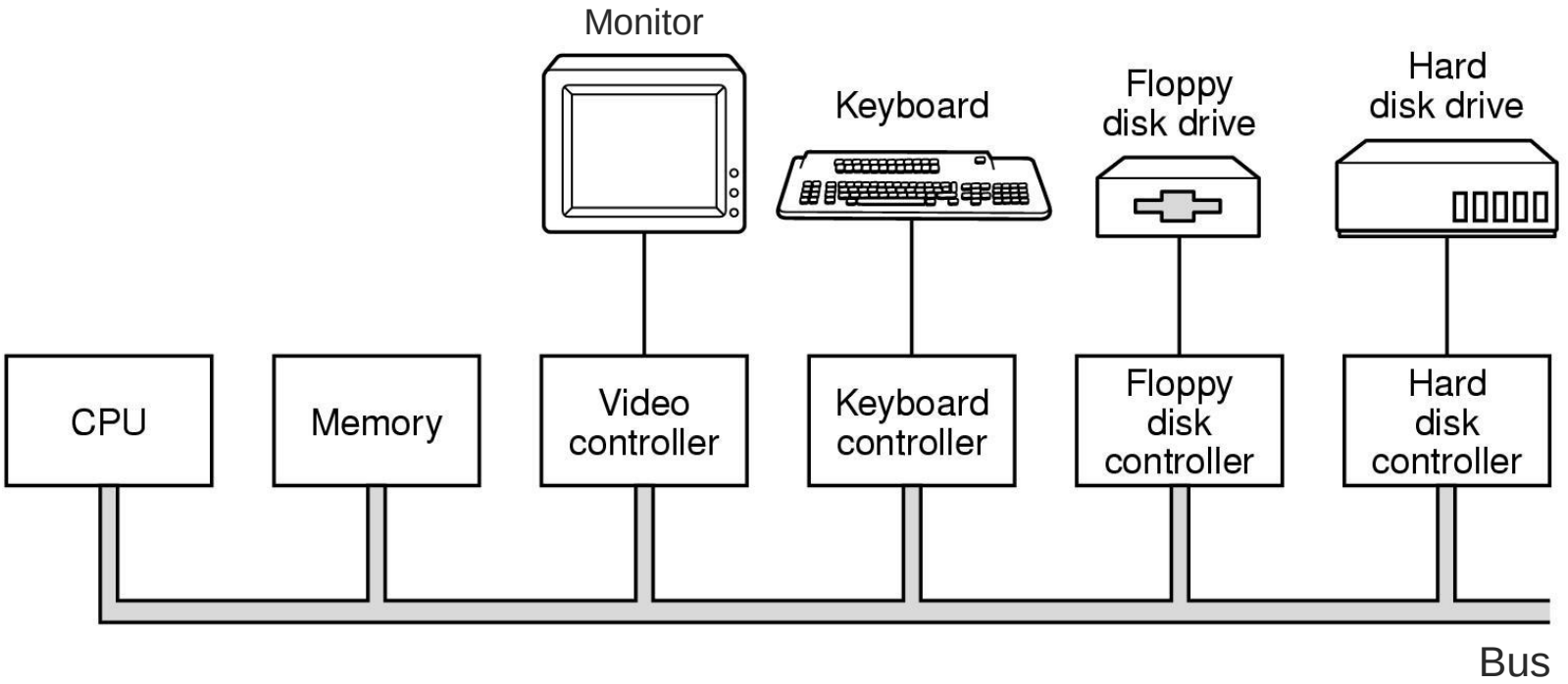
- Structure of a typical job
  - 2nd generation

History of Operating Systems



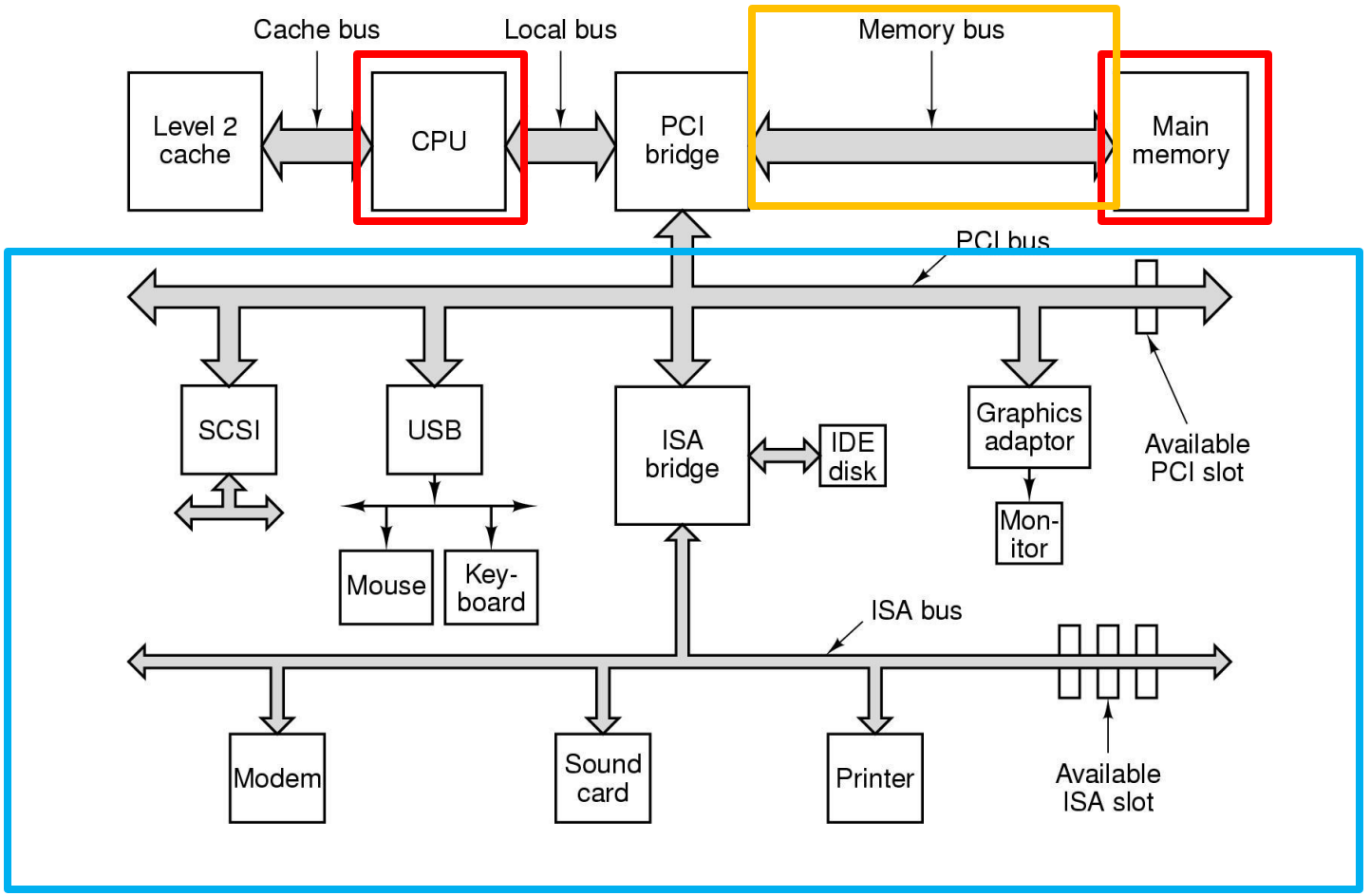
- Multiprogramming/timesharing system
  - Three jobs in memory – 3rd generation

Computer Hardware Review

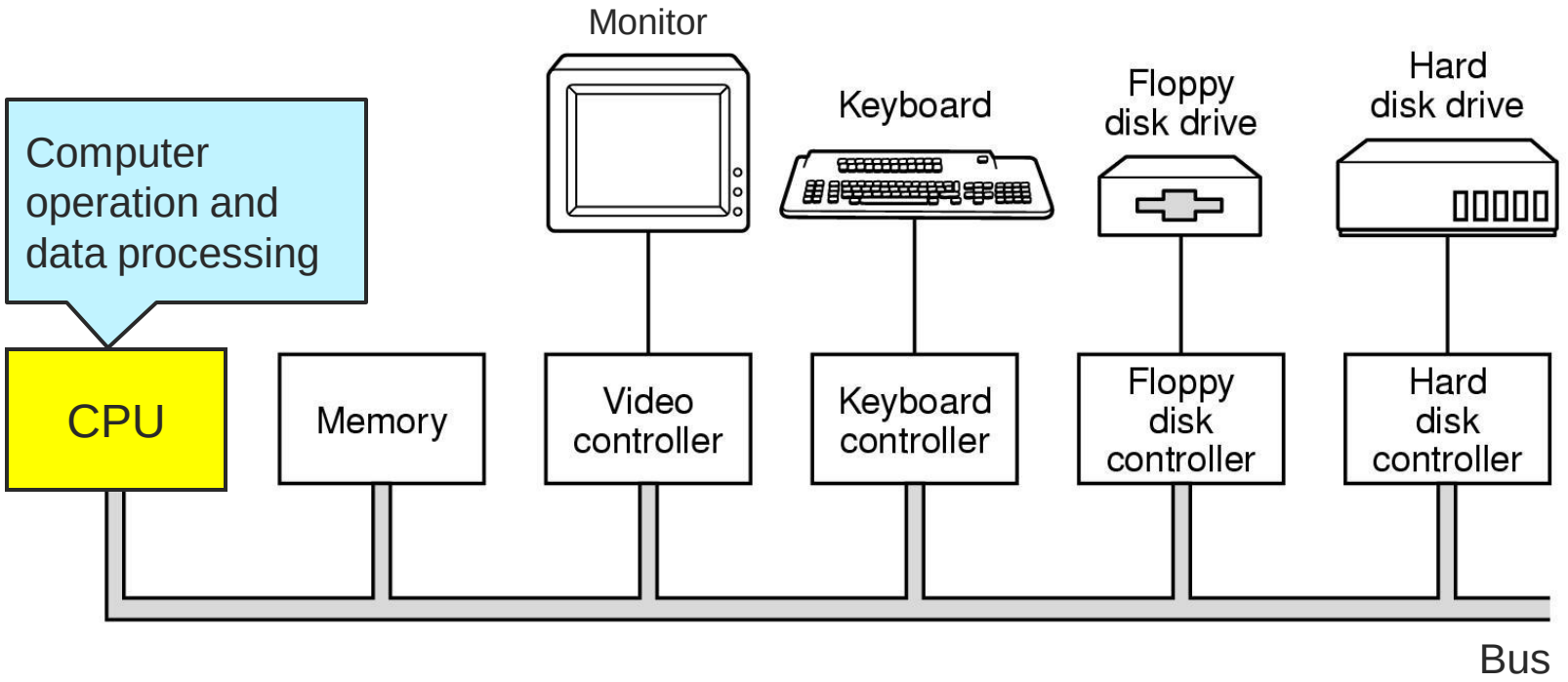


- Components of a simple personal computer

Early Pentium system



Computer Hardware Review



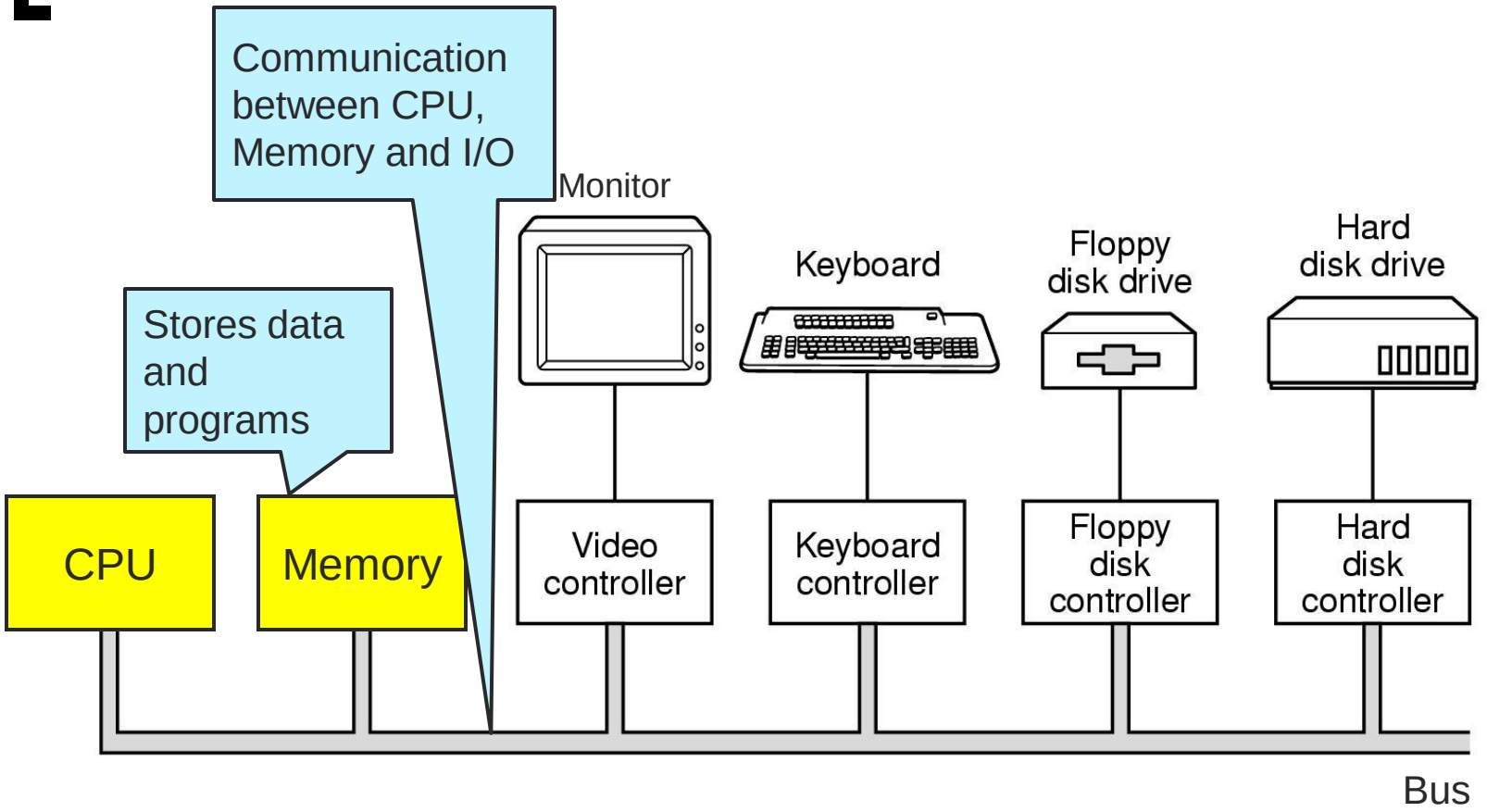
- Components of a simple personal computer

CPU , From CS231

- Fetch instruction from code memory
- Fetch operands from data memory
- Perform operation (and store result)
- (Check interrupt line)
- Go to next instruction

- 'Conventional CPU'
  (Ignore pipeline, optimization complexities)

Computer Hardware Review



- Components of a simple personal computer

CPU Registers

- Fetch instruction from code memory
- Fetch operands from data memory
- Perform operation (and store result)
- Go to next instruction

- Note: CPU must maintain certain state
  - Current instructions to fetch (program counter)
  - Location of code memory segment
  - Location of data memory segment

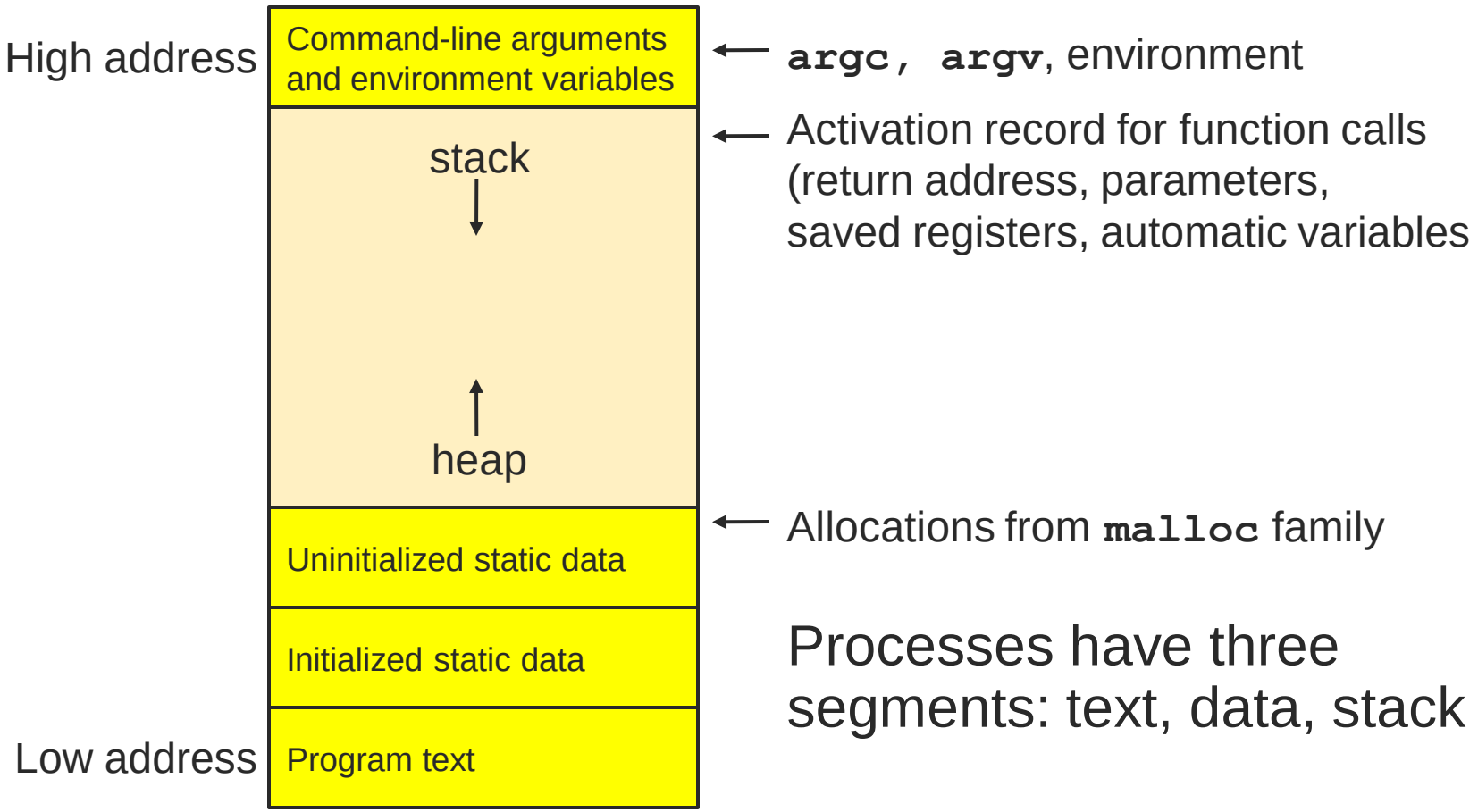
CPU Register Examples

- Hold instruction operands
- Point to start of
  - Code segment
  - Data segment
  - Stack segment
- Point to current position of
  - Instruction pointer
  - Stack pointer

CPU Register Examples

- Hold instruction operands
- Point to start of
  - Code segment
  - Data segment
  - Stack segment
- Point to current position of
  - Instruction pointer
  - Stack pointer
    - Why stack?

Sample Layout for program image in main memory

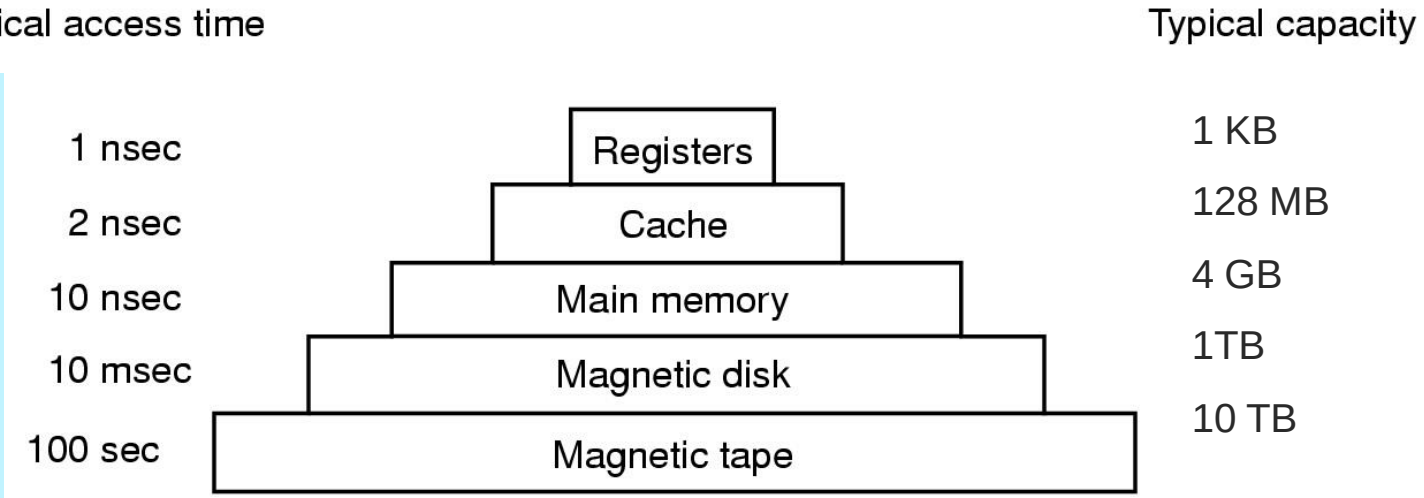


Processes have three segments: text, data, stack

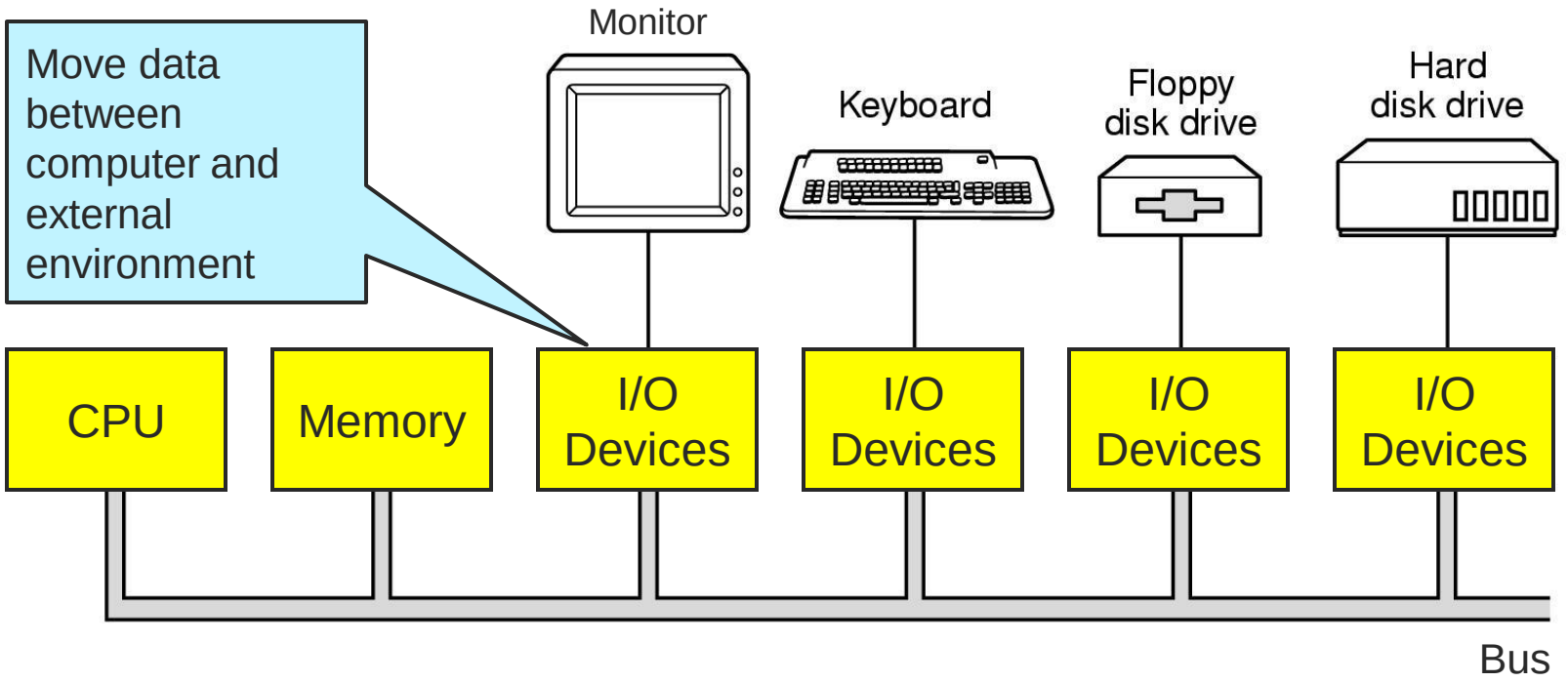
Memory Hierarchy

- Locality of reference

1. Decreasing cost per bit
2. Increasing capacity
3. Increasing access time
4. Decreasing frequency of access

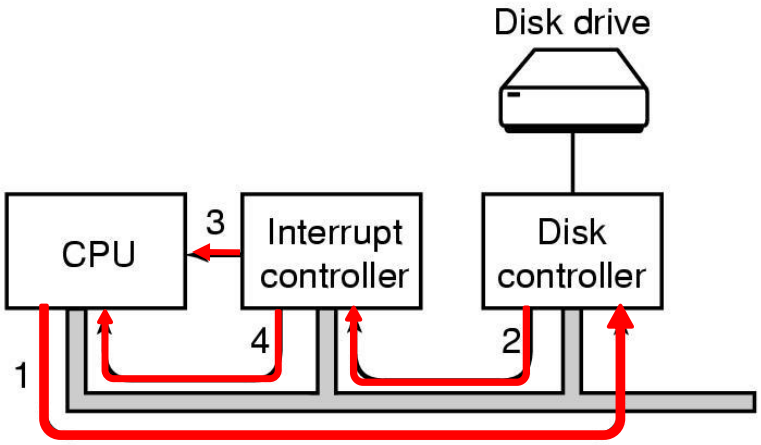


Computer Hardware Review

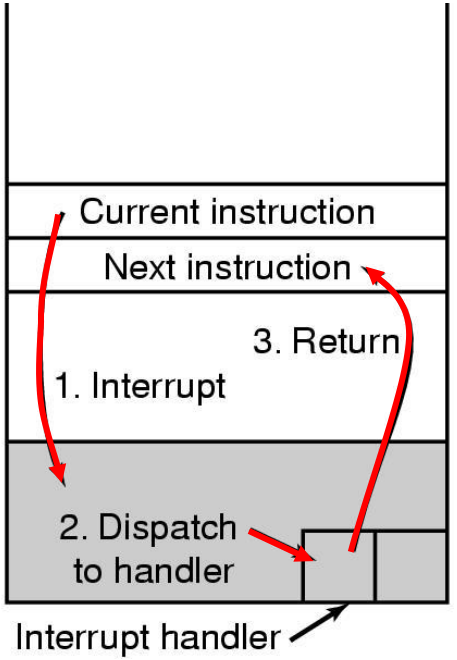


- Components of a simple personal computer

I/O Interrupt Mechanism



(a)

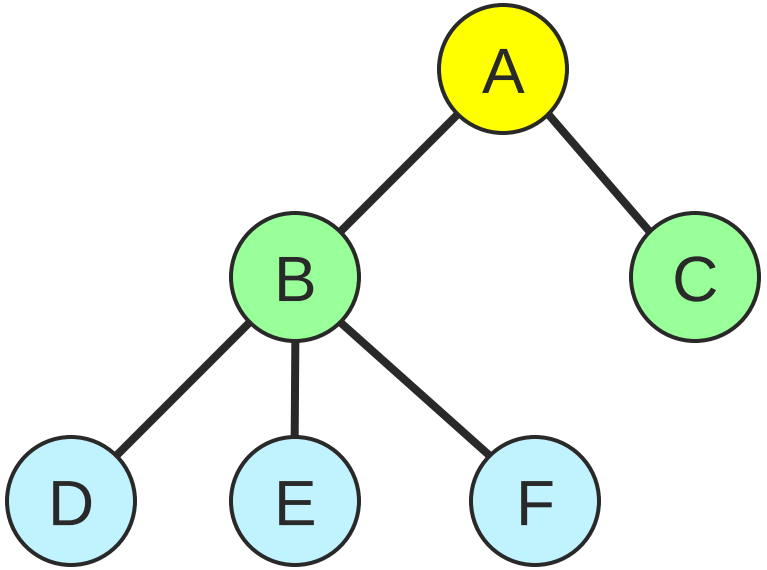


(b)

- Steps in starting an I/O device and getting interrupt
- How the CPU is interrupted

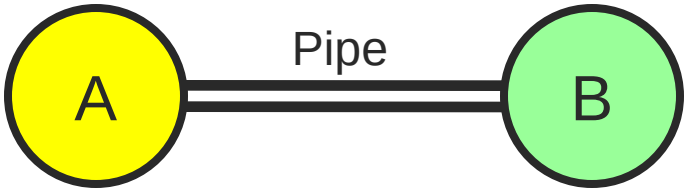
Operating System Concepts

- Process
  - An executable instance of a program
  - Only one process can use the CPU at a time
- A process tree
  - A created two child processes, B and C
  - B created three child processes, D, E, and F

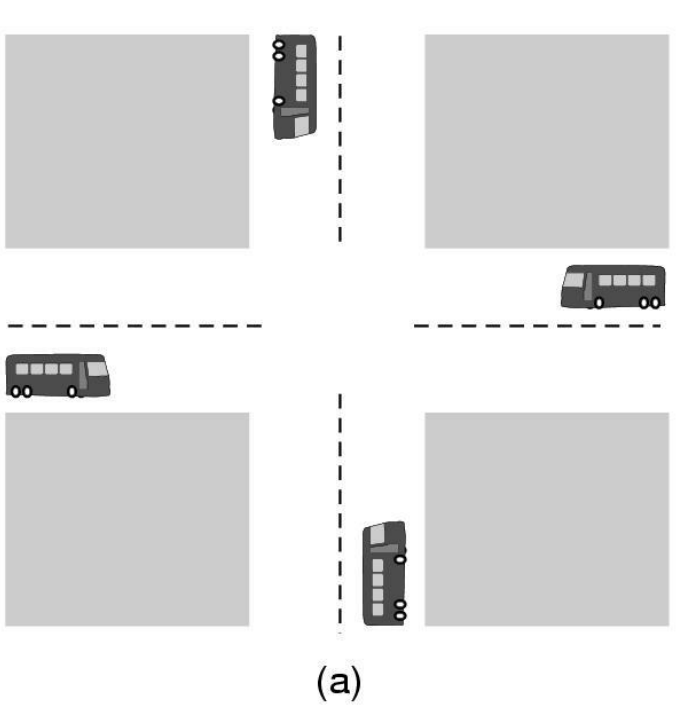


Operating System Concepts

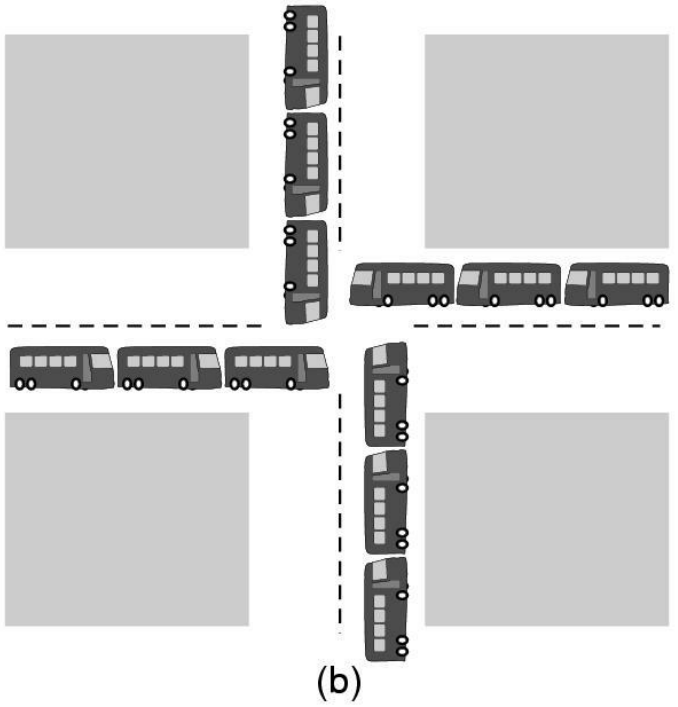
- Context Switching
  - How would you switch CPU execution from one process to another?
- Semaphores
  - Control access to resources
- Inter-process Communication
  - Two processes connected by a data/control pipe



Shared Resources, Conflicts, and Deadlocks



(a) A potential deadlock



(b) An actual deadlock

System Times

Item	Time	Scaled Time in Human Terms (2 billion times slower)
Processor cycle	0.5 ns (2 GHz)	1 s
Cache access	1 ns (1 GHz)	2 s
Memory access	15 ns	30 s
Context switch	5,000 ns (5 micros)	167 m
Disk access	7,000,000 ns (7 ms)	162 days
System quanta	100,000,000 (100 ms)	6.3 years

Summary

- Resource Manager
- Hardware independence
- Virtual Machine Interface
- POSIX
- Concurrency & Deadlock