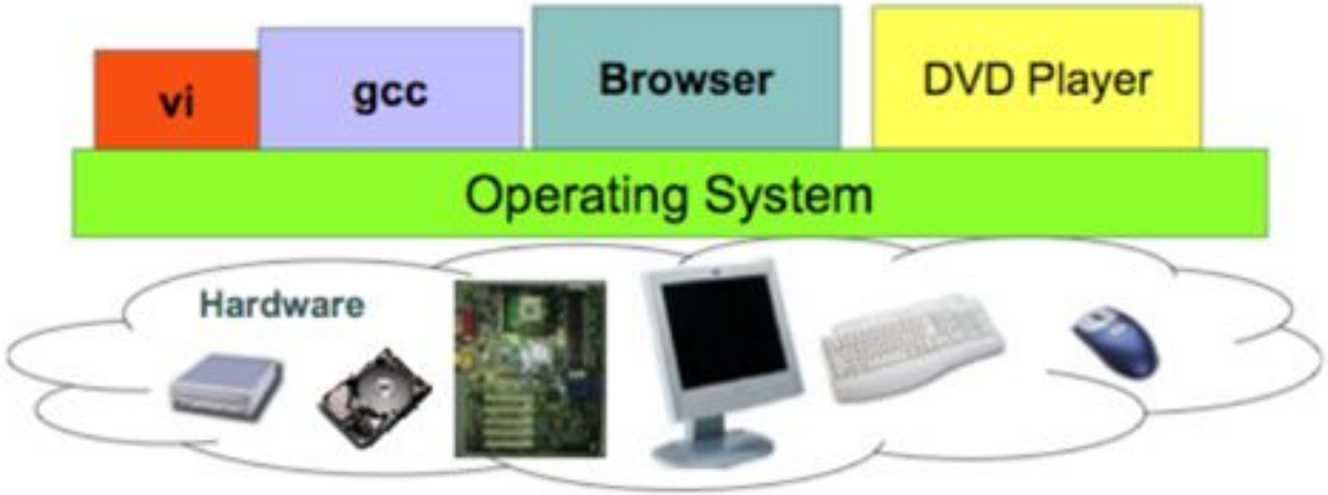
CS100 #14

Operating Systems

Vadim Surov

What is an OS?

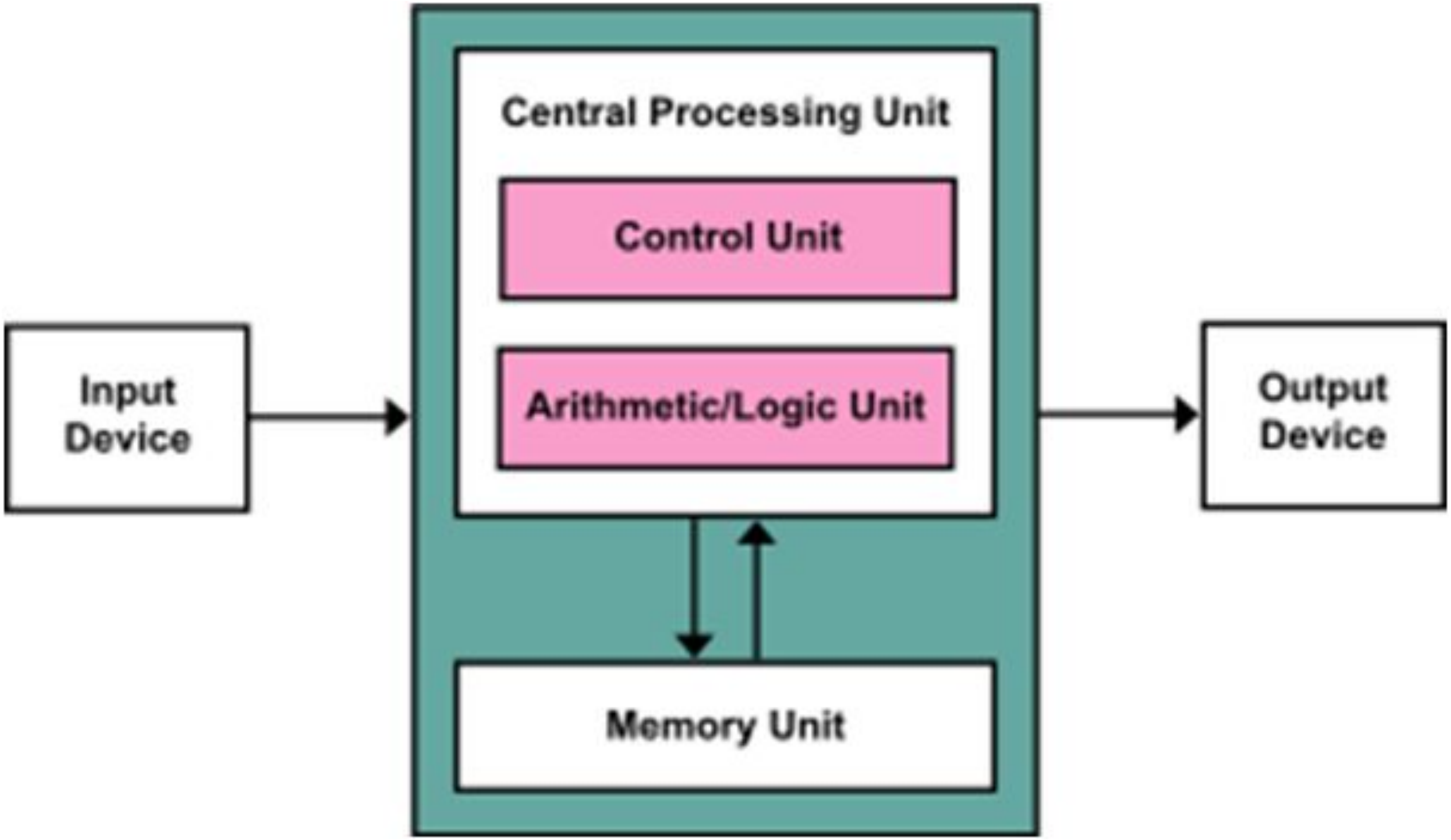


- OS is an interface between your hardware and software.
- It acts as a resource manager

Why we need to study OS?

- OS is a key part of a computer system
- Understand how computers work under the hood
- Combine language, hardware, data structures, and algorithms.
- Write better code
- Write an OS

Recap Computer Architecture

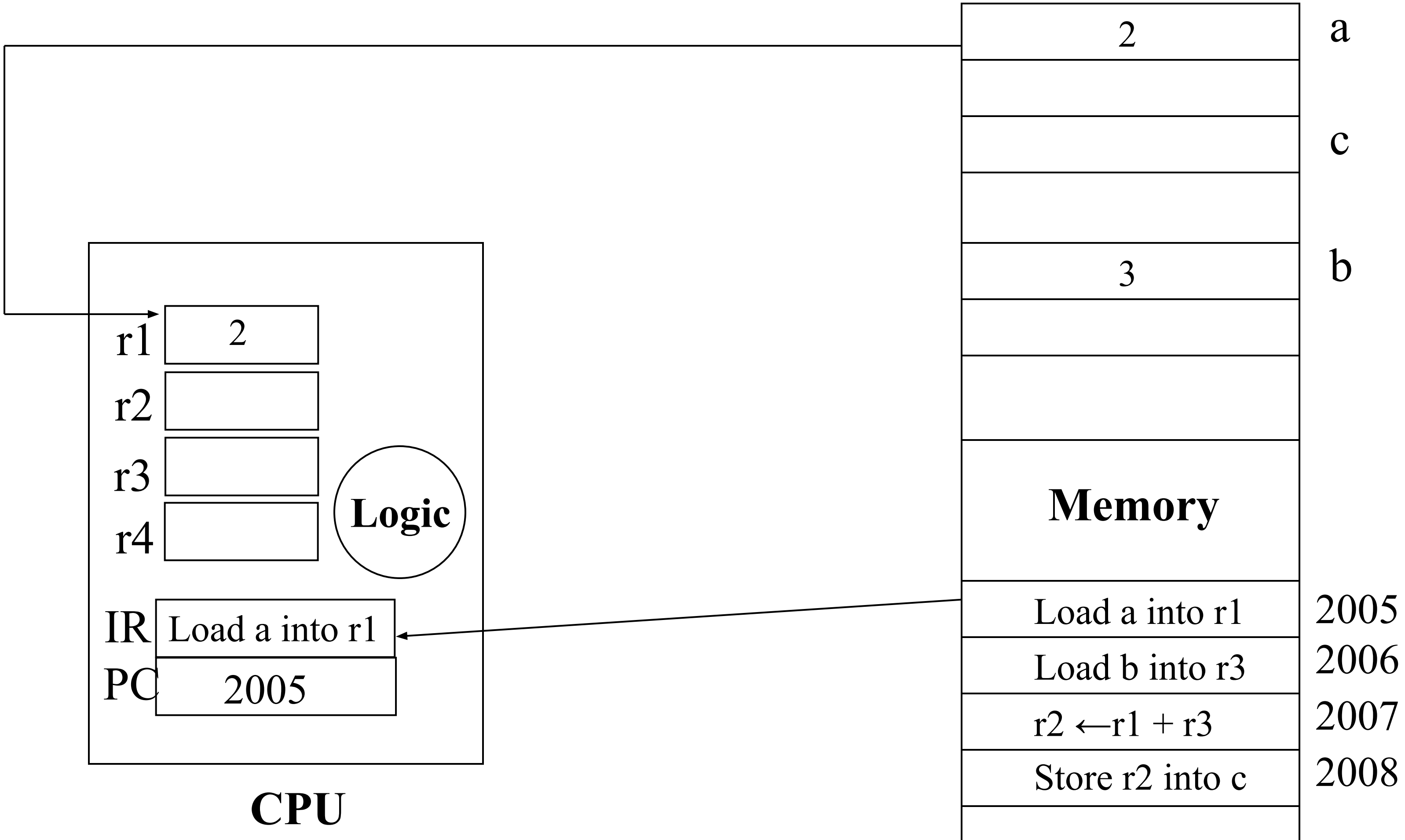


Example Running A Simple Program

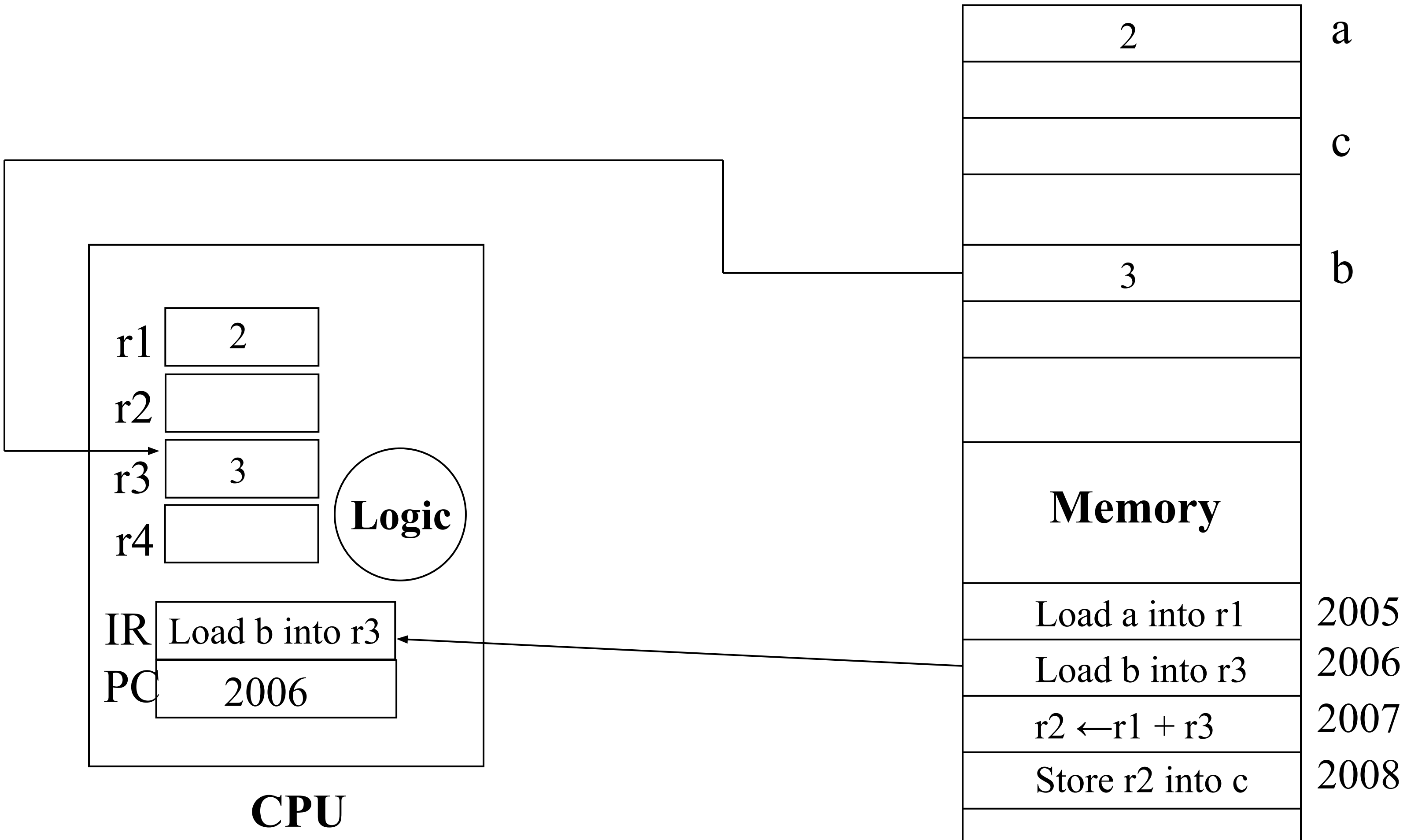
- Want to add values of variables a and b (assumed to be in memory), and put the result in variable c in memory

- Instructions in program
    - Load a into register r1
    - Load b into register r3
    - r2 ← r1 + r3
    - Store r2 in c

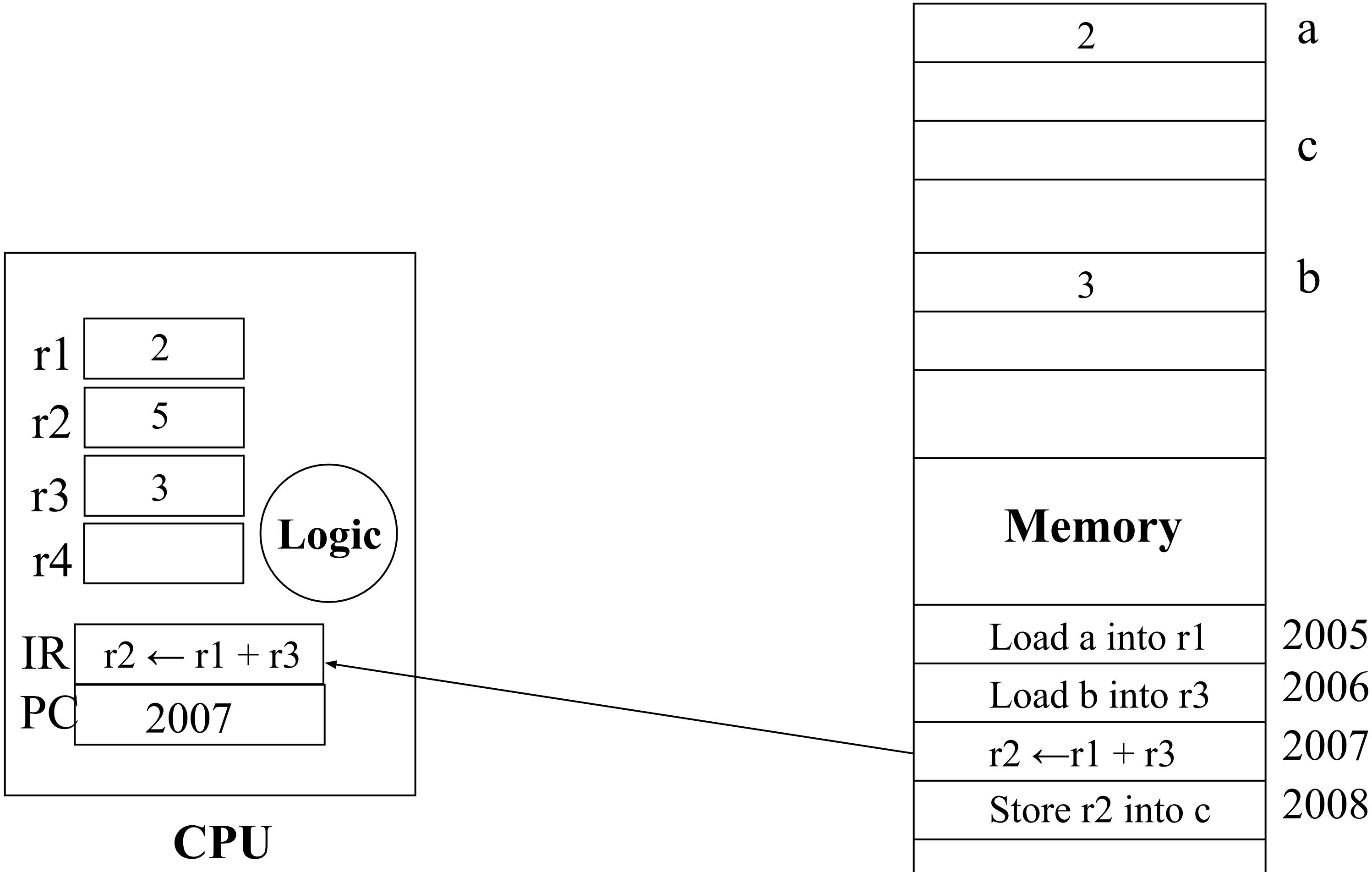
A Simple Program: Running



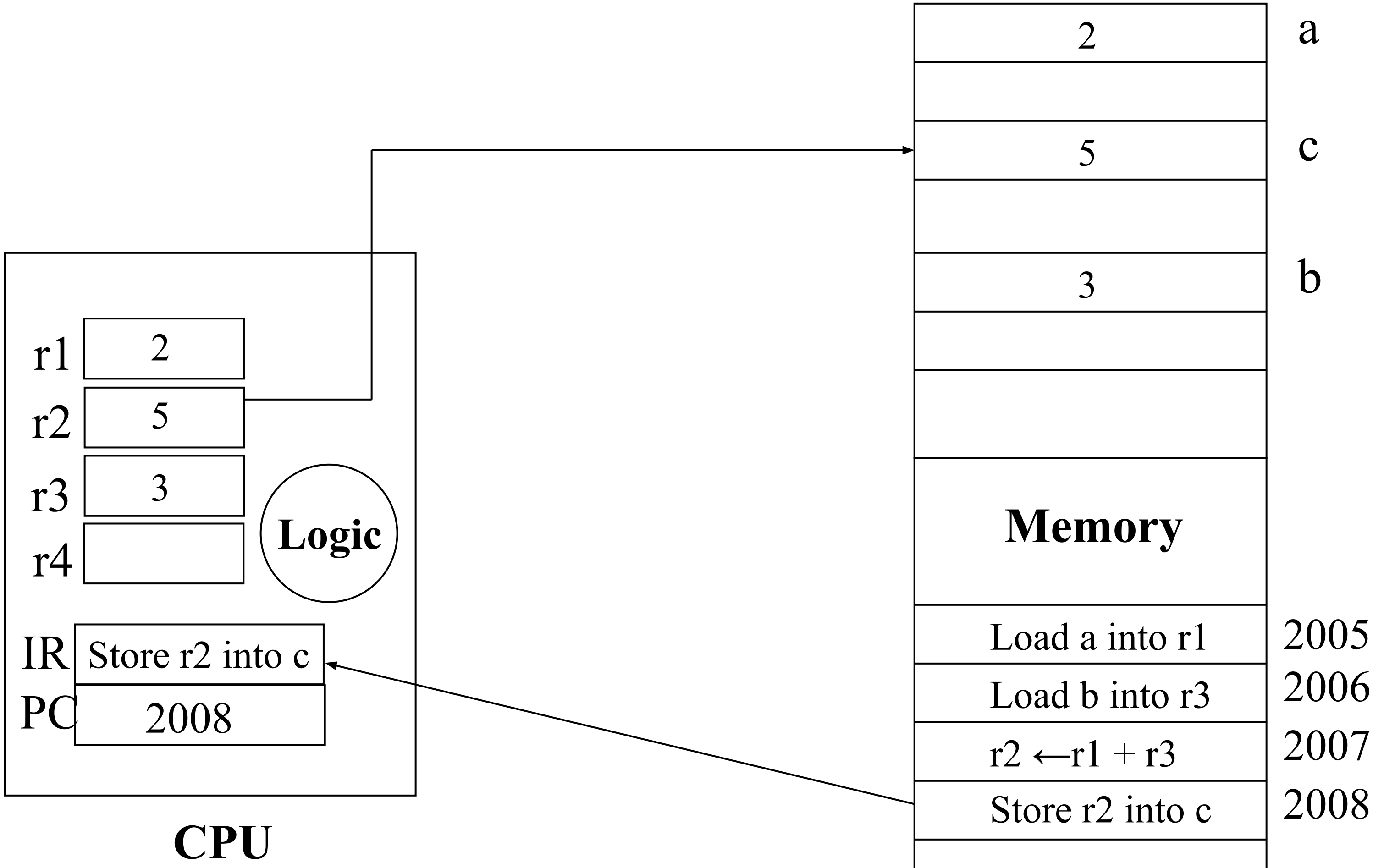
A Simple Program: Running



A Simple Program: Running



A Simple Program: Running



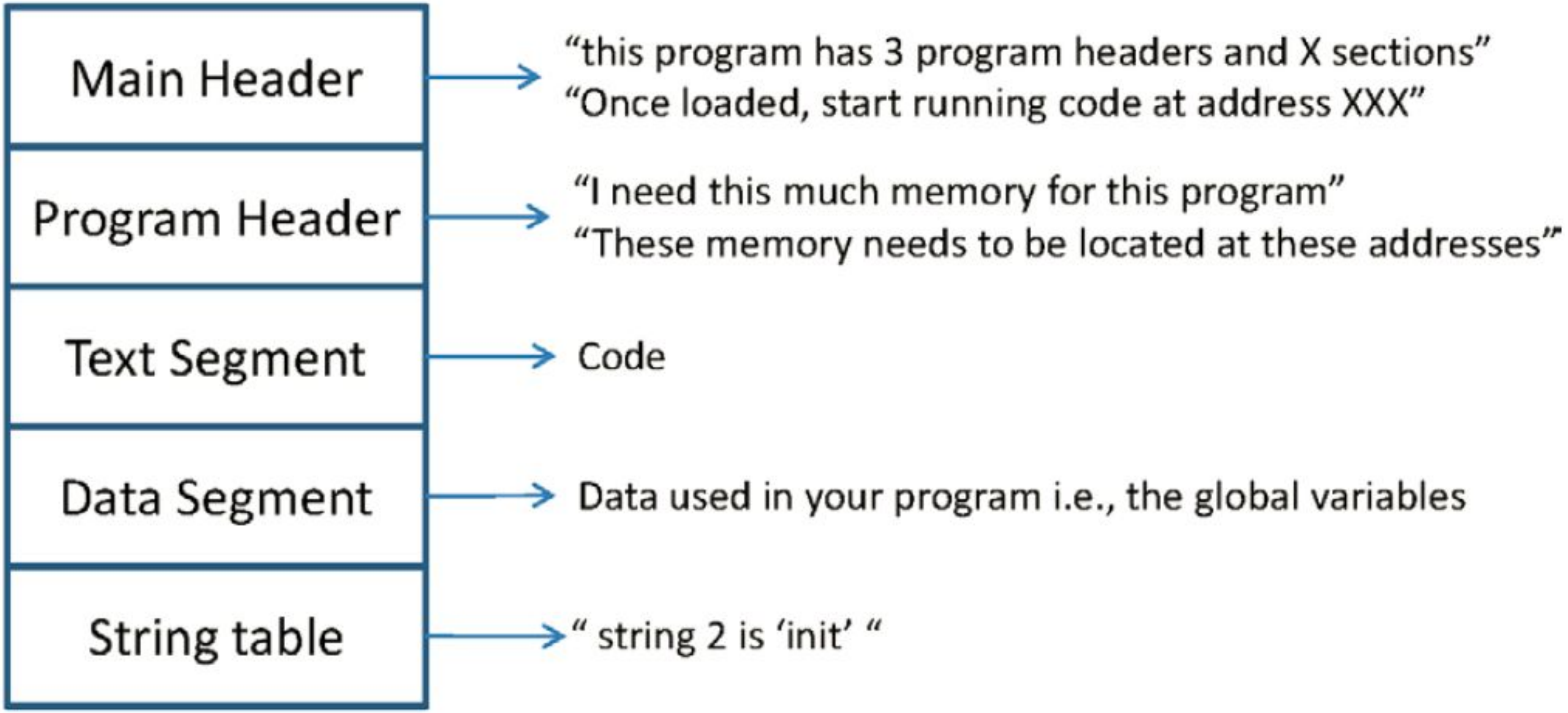
Program execution

- So how does a program execute?
- Naïve answer:
  - load the program into memory and point the PC to the start of the program.

Loading a Program

- Need a program called a loader
  - Able to read the executable format
  - Copy the text and data segments into the correct memory addresses.
  - Allocate space for Stack and heap for the new running program
  - Set the Program Counter value to the address of the starting instruction of the loaded program.
  - The newly loaded program runs

Layout of an executable



"this program has 3 program headers and X sections"
"Once loaded, start running code at address XXX"

"I need this much memory for this program"
"These memory needs to be located at these addresses"

Code

Data used in your program i.e., the global variables

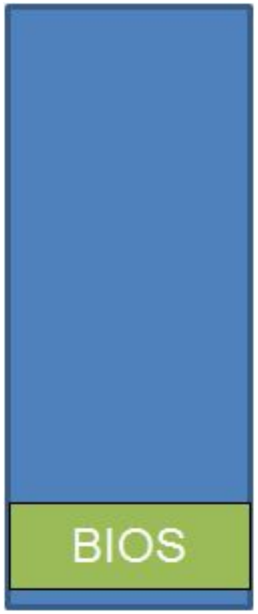
" string 2 is 'init' "

Loader

- Chicken and egg problem
- Who loads the loader?
- Need to talk about the boot sequence

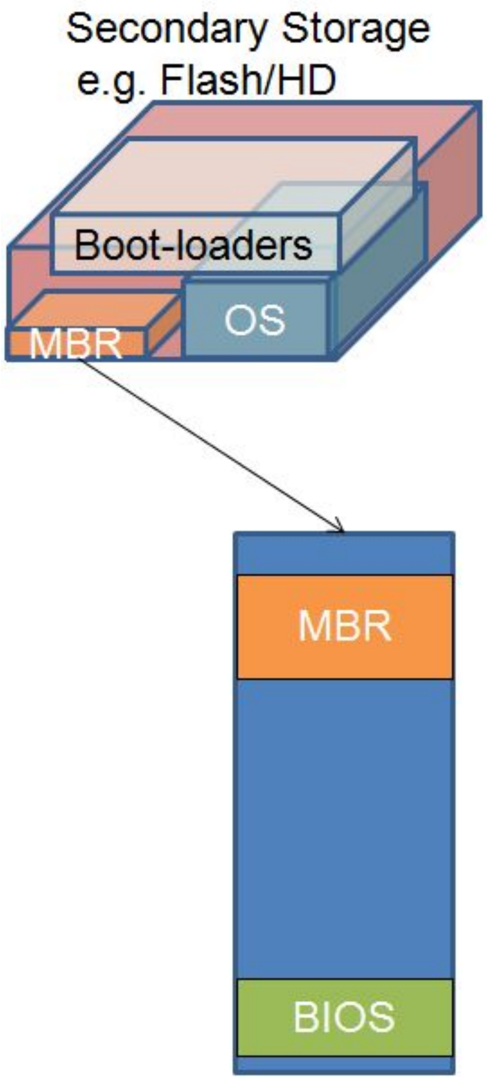
The Boot-up process

1. Power-On
2. Run BIOS
   - The processor is pre-programmed to always look at the same place in the system BIOS ROM for the start of the BIOS boot program located at FFFFFFF0h
   - Performs “power-on self test”
   - Initialize peripherals (video card, etc)
   - Search through the secondary storages i.e., hard disks for bootable drive



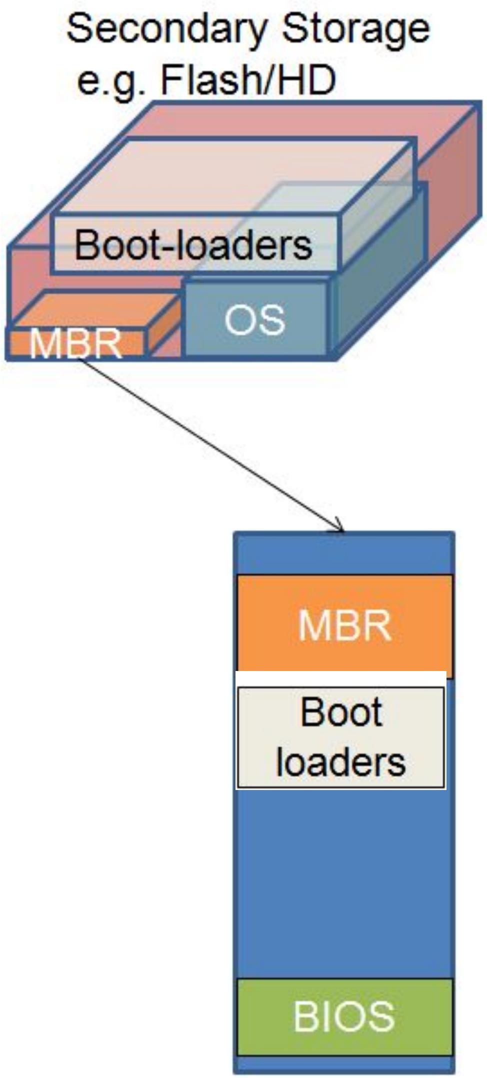
The Boot-up process

1. Power-On
2. Run BIOS
3. Bios load and run MBR
   a. Look for Master Boot Record
   b. Located at 1’st sector of the hard disk
   c. Small 512 bytes



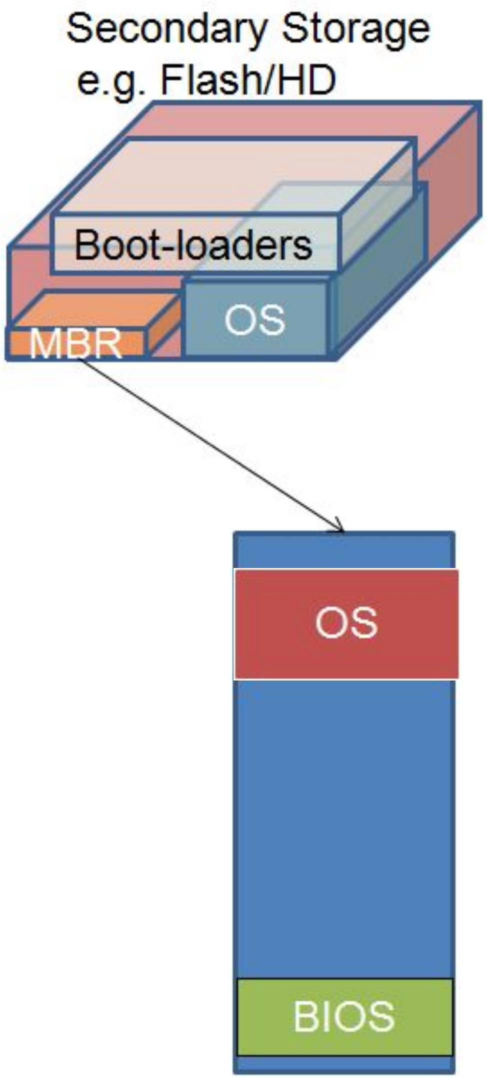
The Boot-up process

1. Power-On
2. Run BIOS
3. Bios load and run MBR
4. MBR loads an OS boot loader program that in turns loads the rest of the OS



The Boot-up process

1. Power-On
2. Run BIOS
3. Bios load and run MBR
4. MBR loads an OS boot loader
5. Load and Run OS



What does an OS do?

- Interface/Abstraction
  - API for programmers
  - Remove need for low-level details
- Portability
- Resource Management
  - Virtualization
- Security

OS Types

- There are several architectures which all require a different OS:
  - Desktop PCs
  - Parallel Systems
  - Distributed Systems
  - Clustered Systems
  - Real-time Systems
  - Embedded Systems

Desktop Systems

- Personal Computers – computer system dedicated to a single user.
- I/O devices – keyboards, mice, display screens, small printers.
- User convenience and responsiveness.
- Can adopt technology developed for larger operating system
- Often individuals have sole use of computer and do not need advanced CPU utilization of protection features.
- May run several different types of operating systems (Windows, MacOS, UNIX, Linux)

Parallel Systems

- Multiprocessor systems with more than one CPU in close communication.
- Tightly coupled system – processors share memory and the internal clock; communication usually takes place through the shared memory.
- Advantages of parallel system:
    - Increased throughput (productivity)
    - Economical
    - Increased reliability
        - graceful degradation
        - fail-soft systems

Parallel Systems

- Symmetric multiprocessing (SMP)
  - Each processor runs an identical copy of the operating system.
  - The OS code is usually shared.
  - Many processes can run at once without performance deterioration.
  - Most modern operating systems have SMP support.
  - OS has to cater for protection of data.

Parallel Systems

- Asymmetric multiprocessing
  - Each processor is assigned a specific task; master processor schedules and farms work to slave processors.
  - More common in extremely large systems like mainframes with hundreds of processors.

Distributed Systems

- Distribute the computation among several physical processors.
- Loosely coupled system – each processor has its own local memory; processors communicate with one another through various communications lines, such as high-speed buses or network communication.

Distributed Systems

- Advantages of distributed systems.
  - Resources Sharing
  - Computation speed up – load balancing
  - Scalability
  - Reliability
  - Fail-Safe
  - Communications
- May make use of commodity platforms.
- OS has to cater for resource sharing.
- May be either client-server or peer-to-peer systems.

Clustered Systems

- Clustering allows two or more systems to share storage.
- Provides high reliability.
- Asymmetric clustering: one server runs the application while other servers standby.
- Symmetric clustering: all N hosts are running the application.
- Used mainly for database applications where a file server exists.

Real-Time Systems

- Often used as a control device in a dedicated application such as controlling scientific experiments, medical imaging systems, industrial control systems, and some display systems.
- Well-defined fixed-time constraints.

Embedded Systems

- Personal Digital Assistants (PDAs)
- Cellular telephones
- Issues:
  - Limited memory
  - Slow processors
  - Small display screens.
- Usually most features of typical OS's are not included at the expense of the developer.
- Emphasis is on I/O operations.
- Memory Management and Protection features are usually absent.