Operating Systems
Introduction

Yanyan Zhuang

Department of Computer Science

http://www.cs.uccs.edu/~yzhuang

Intro of Intro

- Yanyan Zhuang
  - PhD, 2012 yzhuang@uccs.edu
- TA
  - Marwan Alharbi, malharb2@uccs.edu
  - Office hr: Thursdays 1:30-2:30pm on Teams (link in syllabus and Canvas)
- Let's quickly go over the syllabus

Lectures, Assignments, Projects, Exams

- Lectures
  - Monday and Wednesday
  - Quizzes (ungraded)
- Homework
  - 3-4 written assignments: individual
- Projects
  - 4 programming assignments (in VMs)
  - Team up (2-3 people) or individual (changing teams allowed)
- Exams (one letter-sized cheat sheet, both sides)
  - Midterm: in class
  - Final: 5/14/2025 https://shorturl.at/tRcuh

Projects are done on VMs

- VM setup
  - Don't wait till last minute to work/submit..
- **Do not create more than 2 VMs**
  - Limited IP addresses available to the class
  - If you'd like to abandon a VM, rename it "delete-me"
    - Not auto deleted – I will request IT to delete every now and then
  - Sharing a VM among team members is encouraged
  - Power off the VM when not in use

Late Policy

- Days late / percent off
  - 1-6days / -10% each day
  - 6+days / -100%
- Final submission
  - Midnight 5/10
  - All unsubmitted assignments/projects must be submitted

Where to get help?

- Ask questions in class
- Read slides
- Office hours

Outline

- Why study Operating Systems?
- What will you learn?
- OS overview

Textbook readings: Section 1.1 – 1.2

Why Study Operating Systems?

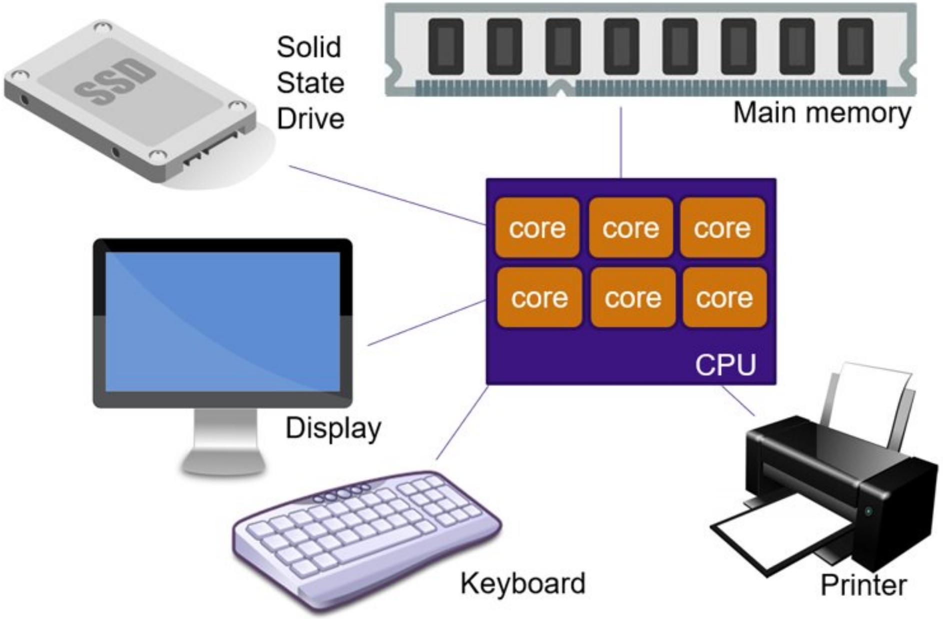
- The most complex software
  - > 27 million lines of code in Linux

Why Study Operating Systems?

- The most complex software
  - > 27 million lines of code in Linux
- The most fundamental software
  - OSes are almost everywhere, e.g., supercomputer, PC, phone…

Hardware Components of a Computer

- One or more processors
- Main memory
- Disks/flash drives
- I/O devices
    - ○ Printer, keyboard, mouse
    - ○ Display, network interface



Why Study Operating Systems?

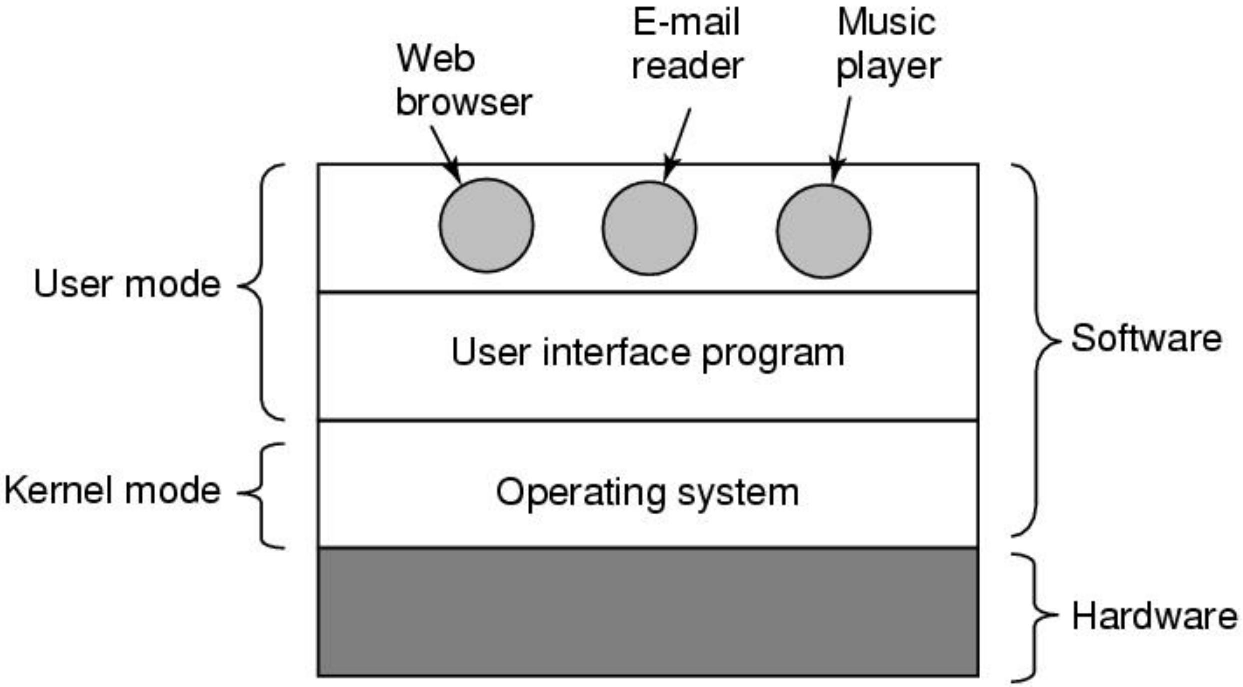
- The most complex software
  - >27 million lines of code in Linux
- The most fundamental software
  - OSes are almost everywhere, e.g., supercomputer, PC, phone…
- By studying OS, you will
  - Learn how computers work
    - Gain a good understanding of OS and hardware
  - Learn about system design
    - Simplicity, portability, performance, and trade-offs

What Will You Learn?

- Hardware abstraction
  - Processes, threads, files …
- Resource management
  - CPU scheduling, memory management, file systems …
- Coordination
  - Multiple programs and users
  - Fairness and efficiency

OS Overview

What is an Operating System?

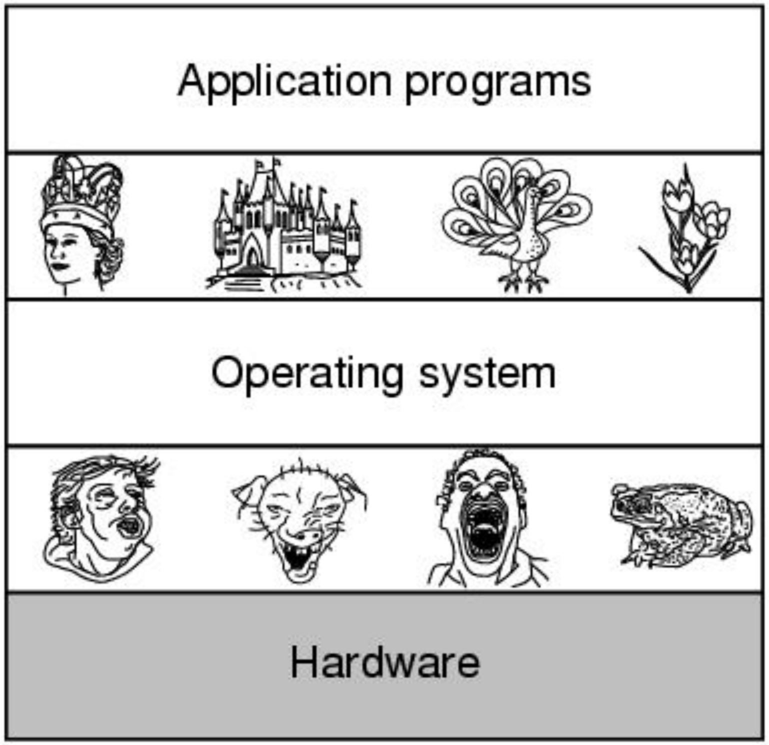


- A computer system consists of
  - hardware
  - system programs
  - application programs

What does an Operating System do?

- It is an extended (or virtual) machine
  - Hides the messy details which must be performed
  - Presents user with a virtual machine, easier to use
  - Abstraction over hardware

The Operating System as an Extended Machine



Beautiful interface

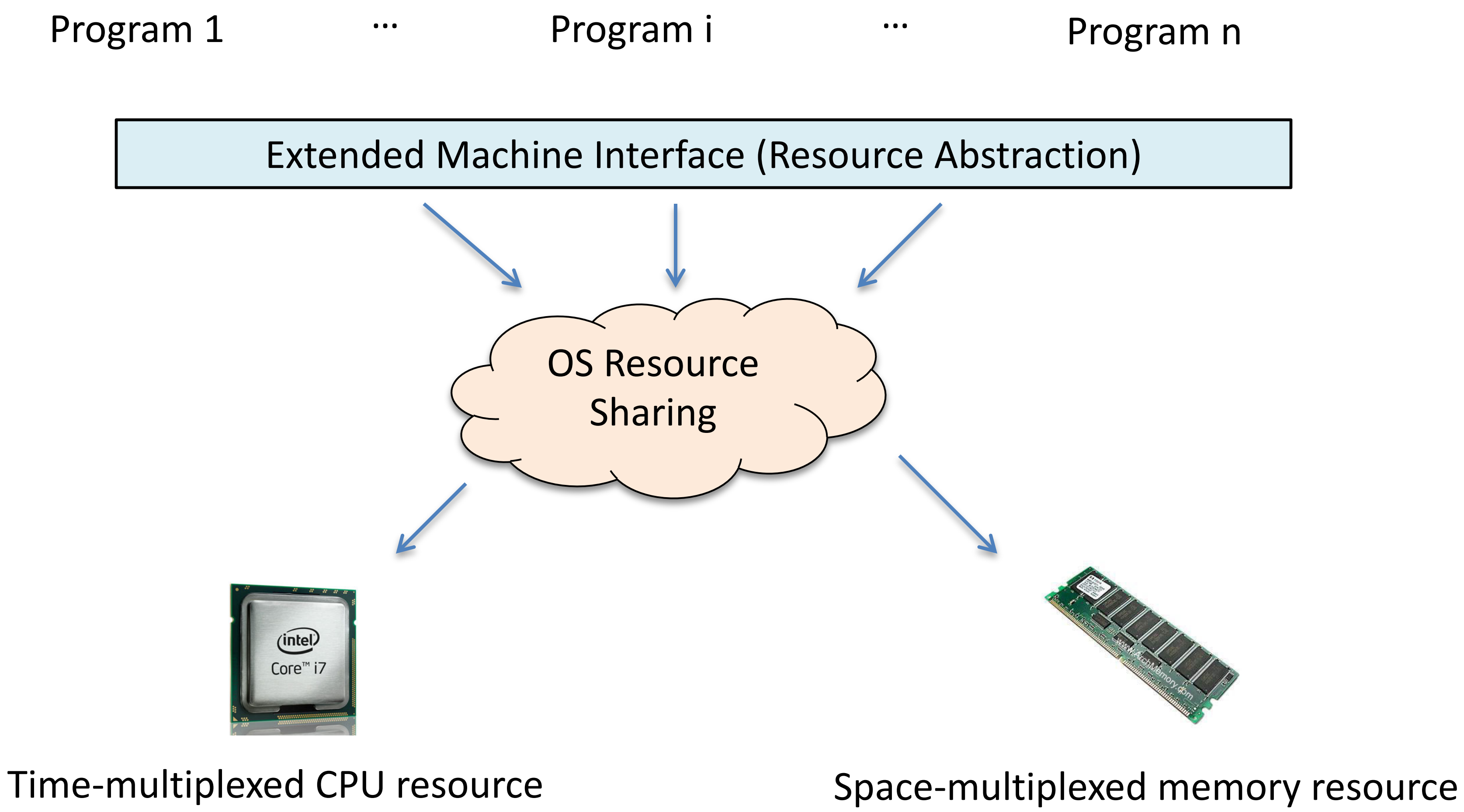
Ugly interface

```
fprintf(fd, “%d”, data);
        ↓
write(fd, buffer, count);
        ↓
file->f_op->write(file, buf,
count, pos);
        ↓
        ⋮
load(block, length, device);
seek(device, track);
out(device, sector);
```

What does an Operating System do?

- It is an extended (or virtual) machine
  - Hides the messy details which must be performed
  - Presents user with a virtual machine, easier to use
  - Abstraction over hardware
- It is a resource manager
  - Protects simultaneous/unsafe usage of resources
  - Ensures fair sharing of resources
    - Types of resources: Time/space multiplexed
    - How: Resource accounting/limiting

The Operating System as a Resource Manager



Program 1 … Program i … Program n

Extended Machine Interface (Resource Abstraction)

OS Resource Sharing

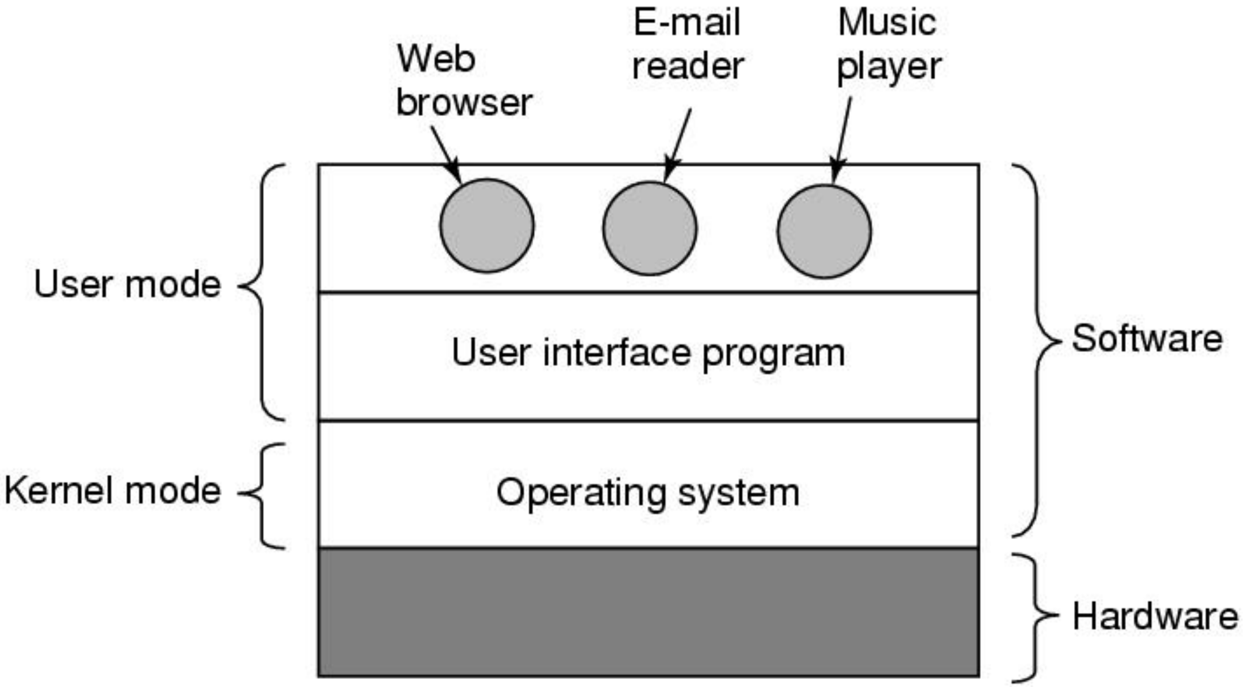
Time-multiplexed CPU resource

Space-multiplexed memory resource

How does an OS work?

- Computers have two modes of operation
  - User mode (application)
  - Kernel mode (OS kernel)
- Transition between user/kernel mode
  - Hardware interrupt – HW device requests OS services (asynchronous/interrupt-driven)
  - System call (aka trap) – user program requests OS services (synchronous/blocking)
  - Exception – error handling
    - Invalid memory access, invalid permission, etc.

What is an Operating System?



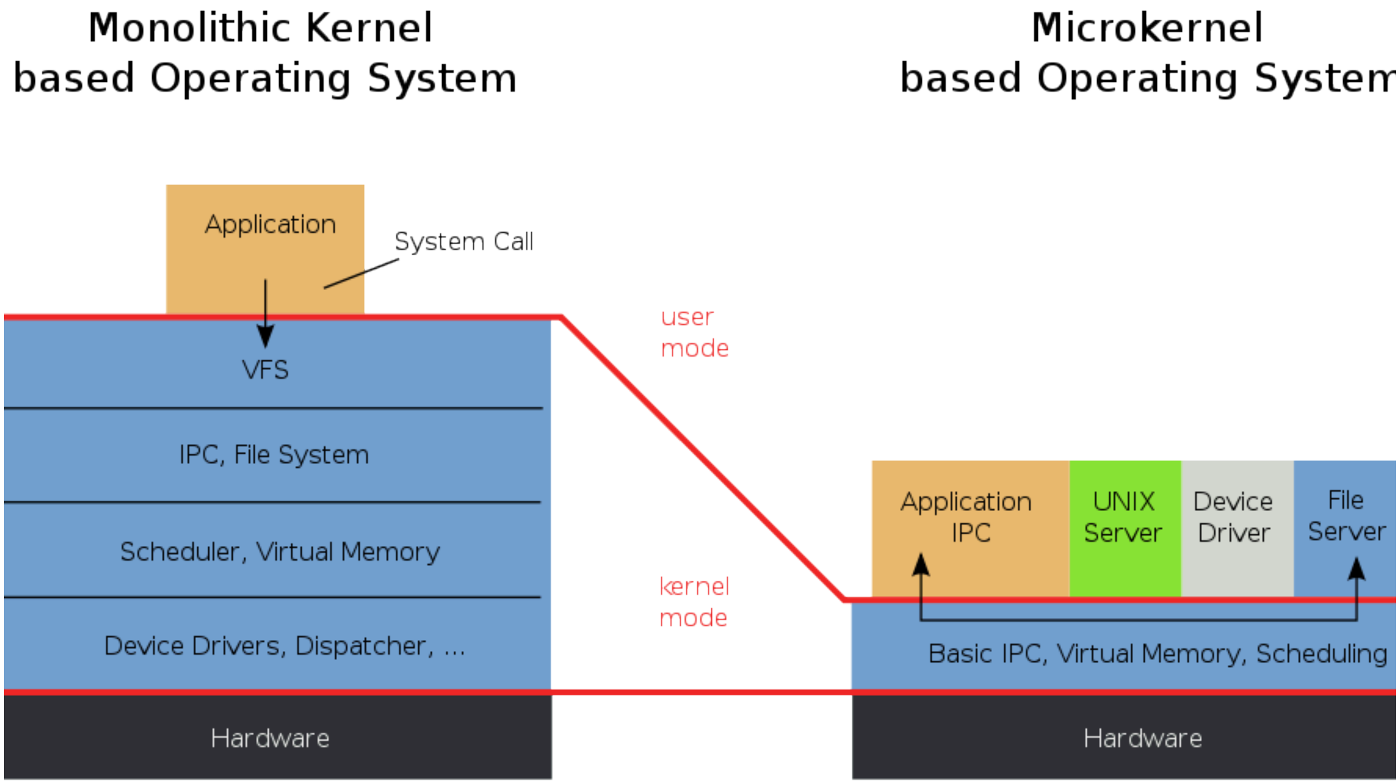
- A computer system consists of
  - hardware
  - system programs
  - application programs

Different Types of OS

- Batch processing
  - Processes jobs one by one
- Time sharing OS
  - Processes multiple jobs in “round robin”
- Real-time OS
  - Still time-sharing, but has deadlines for certain jobs
- Distributed OS
  - Multiple computers run a single copy of OS
- Embedded OS
  - Runs on cell phones, PDAs, tailored and highly efficient

The Structure of OS

Hybrid Kernel


Monolithic Kernel based Operating System
Microkernel based Operating System
Application
System Call
VFS
IPC, File System
Scheduler, Virtual Memory
Device Drivers, Dispatcher, ...
Hardware
user mode
kernel mode
Application IPC
UNIX Server
Device Driver
File Server
Basic IPC, Virtual Memory, Scheduling
Hardware

UNIX, Linux, Windows 98

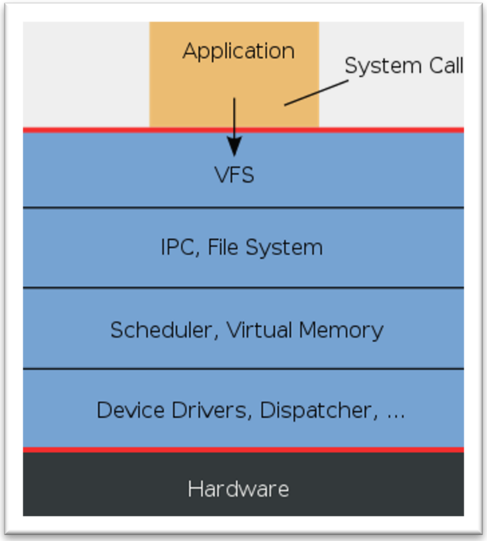
Mach

Windows NT, Mac OS X

Advantage v.s. disadvantage?

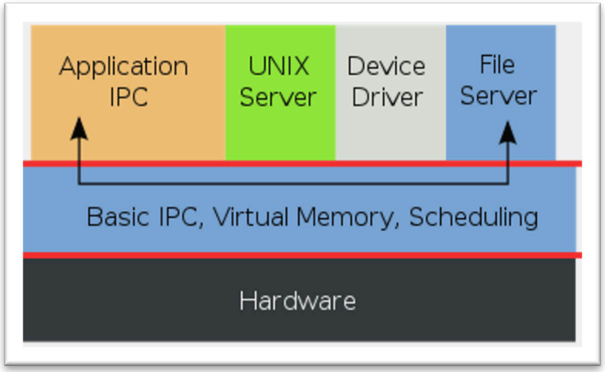
The Structure of OS

- Monolithic kernel
  - A large process running entirely in a **single address space** (a single binary file)
  - All kernel services execute in kernel address space
  - Pros: fast
  - Cons
    - Huge kernel, harder to maintain
    - No protection between kernel components
    - Complex dependencies among components, not easily extensible



The Structure of OS

- Microkernel
  - Kernel broken down to separate processes (aka servers)
  - Servers kept separate and run in different address spaces
  - Communication is done via message passing
    - Servers communicate through IPC (Inter-process Communication)
  - Pros
    - Modular design, easily extensible
    - Easy to maintain
    - More reliable and secure

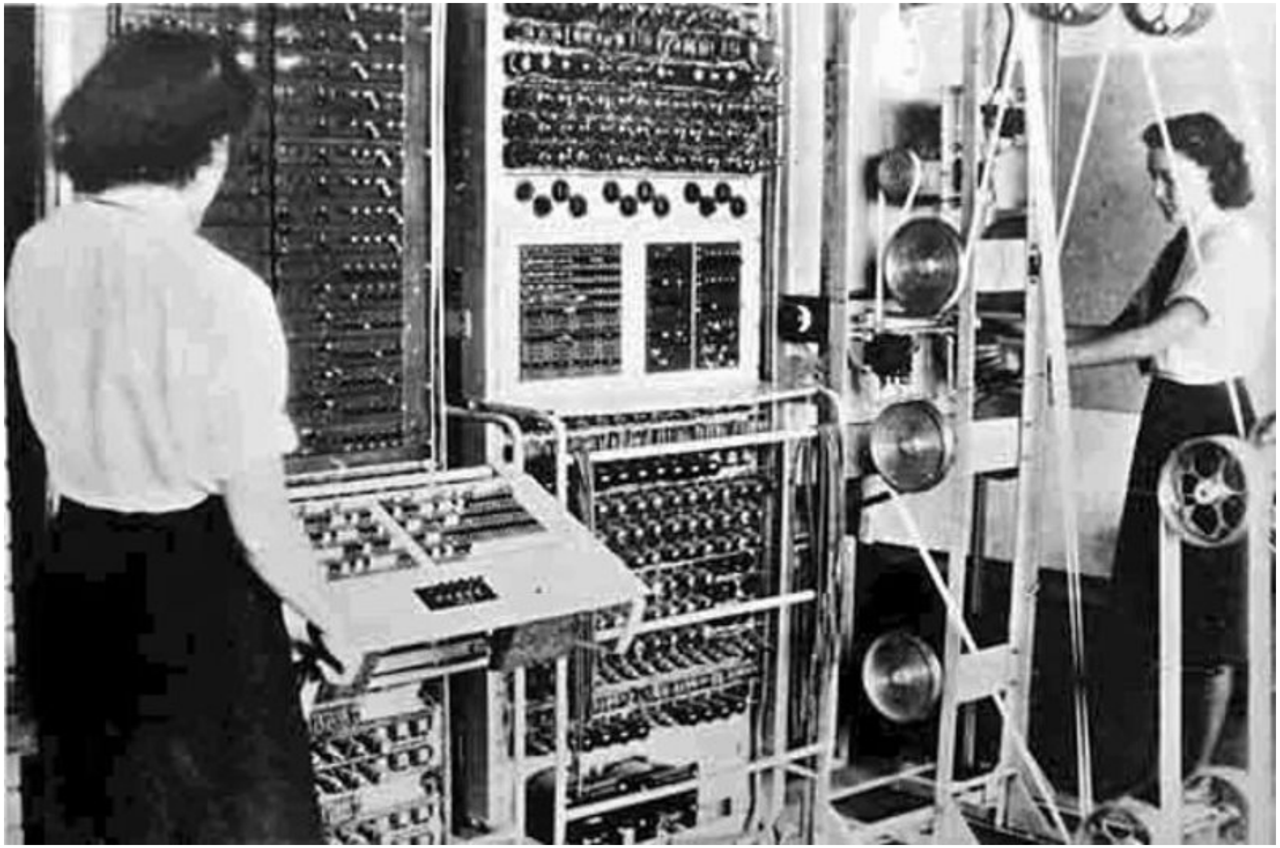


  - Cons: Performance loss, complicated process management

History of Operating Systems

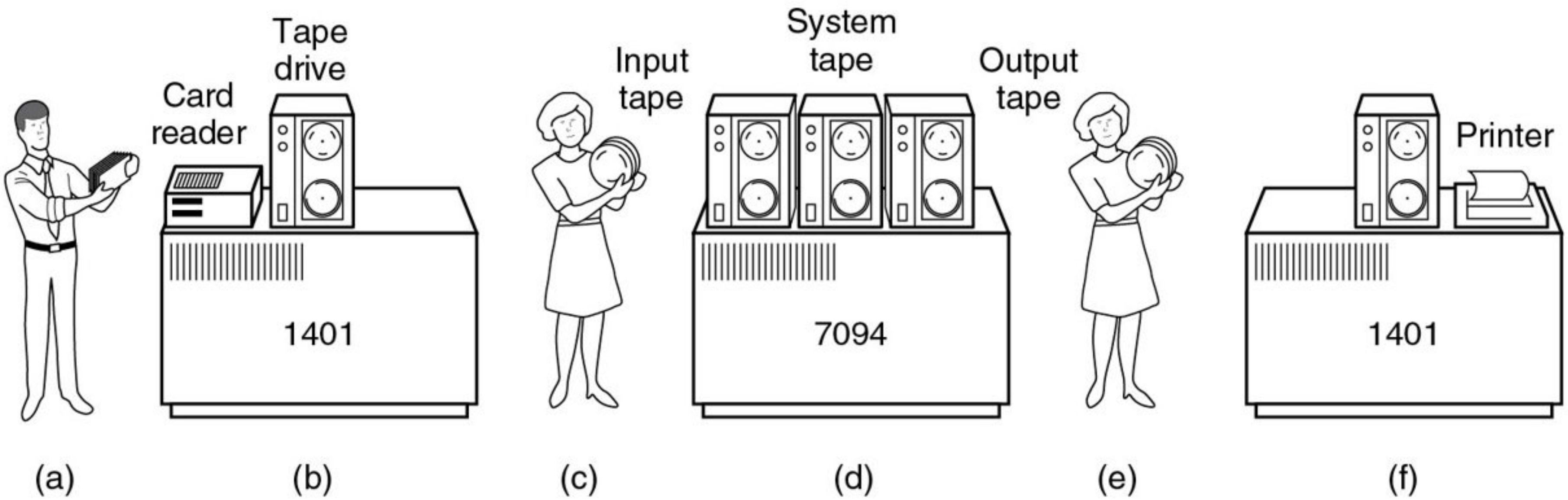
- Coming next…

First Generation: Vacuum tubes (1945-55)



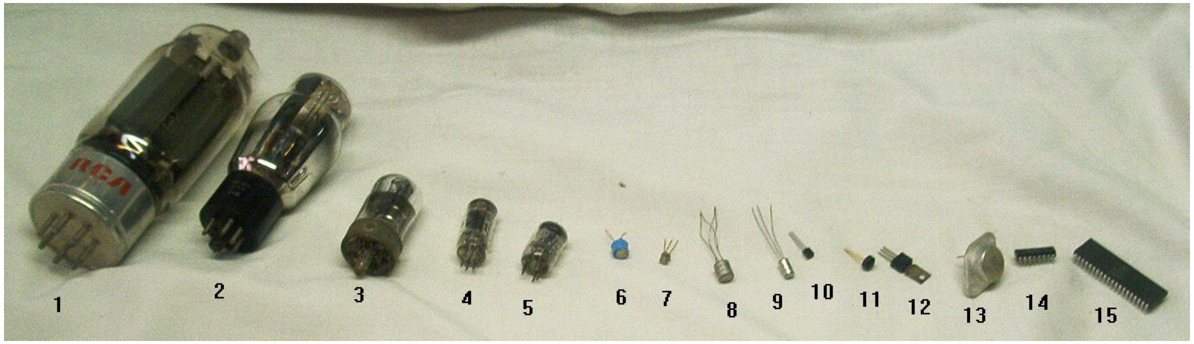
The Colossus Mark 2 Computer
(Anyone knows the first bug?)

Second Gen: Transistors & Batch Systems

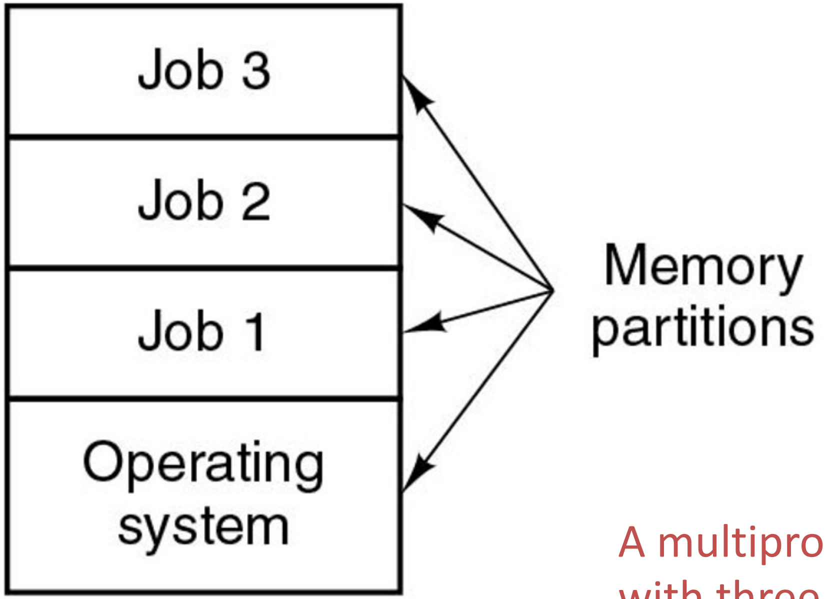


An early batch system:
(a) Programmers bring cards to 1401. (b) 1401 reads batch of jobs onto tape.
(c) Operator carries input tape to 7094. (d) 7094 does computing.
(e) Operator carries output tape to 1401. (f) 1401 prints output.

Third Gen: ICs and Multiprogramming



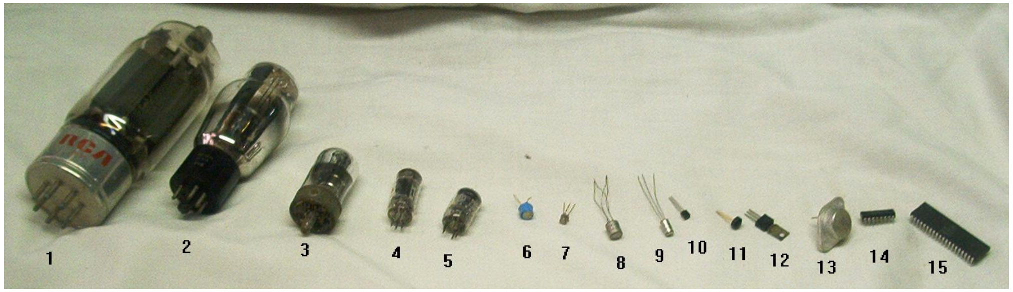
Size comparison: vacuum tubes, transistors, ICs



A multiprogramming system with three jobs in memory

History of Operating Systems

- First generation 1945 - 1955
  - Vacuum tubes
- Second generation 1955 - 1965
  - Transistors and batch systems
- Third generation 1965 – 1980
  - ICs and multiprogramming
- Fourth generation 1980 – present
  - Personal computers: LSI (large scale integration)
- Fifth generation 1990 – present
  - Mobile devices
  - Many-core computers



Summary

- An OS is just a special program
  - Two functionalities: resource abstraction and sharing
  - Provides services to user programs
- Three ways to request OS services
  - Interrupt, system call, and exception
- Next class
  - Overview of computer hardware
  - Organization of operating systems