ECE437/CS481

INTRODUCTION TO OS

OS Development & Evolution

Xiang Sun

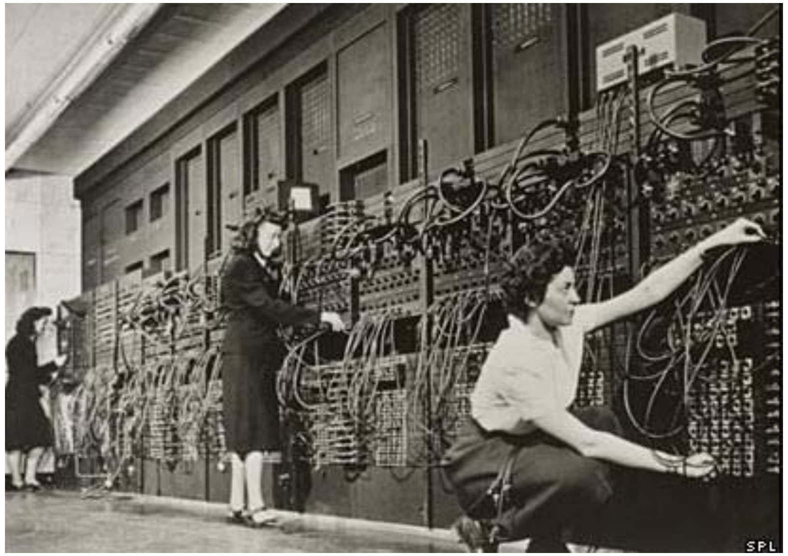
The University of New Mexico

OS/Computer Evolution

- Serial processing (1940s)
  - manual loading and then execution
  - manual operations on a bare machine--punch card, paper tapes, etc.
  - hardware: vacuum tubes
  - problems: inefficient use of the very expensive hardware

- Electronic Numerical Integrator And Computer (ENIAC)

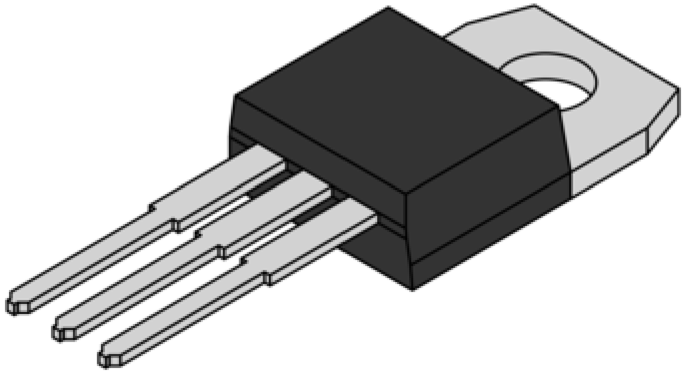
  ---Built at the U of Pennsylvania
  - First general-purpose digital computer.
  - Weighted 30 tons; equipped 18,000 vacuum tubes .
  - A team of five operators working several days on the external wiring.
  - Cost $400,000.
  - No operating system.



OS/Computer Evolution

❑ Serial processing (1950s)

- hardware: vacuum tubes → transistors.
- Program reuse concept:
  - ✓ Programs can be stored.
  - ✓ Programs can be reused as subroutine calls.



❑ The von Neumann Architecture

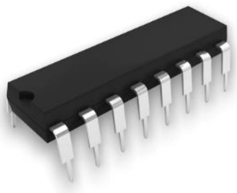
Programs and data could be represented in a similar way and stored in the same internal memory.

❑ IBM dominated the data processing industry

- IBM 7070: first commercial transistorized computers.
- IBM 7090: 1) a 36-bit scientific machine; 2) with IBSYS operating system—a tape-based operating system.

OS/Computer Evolution

❑ Batch processing (1960s)



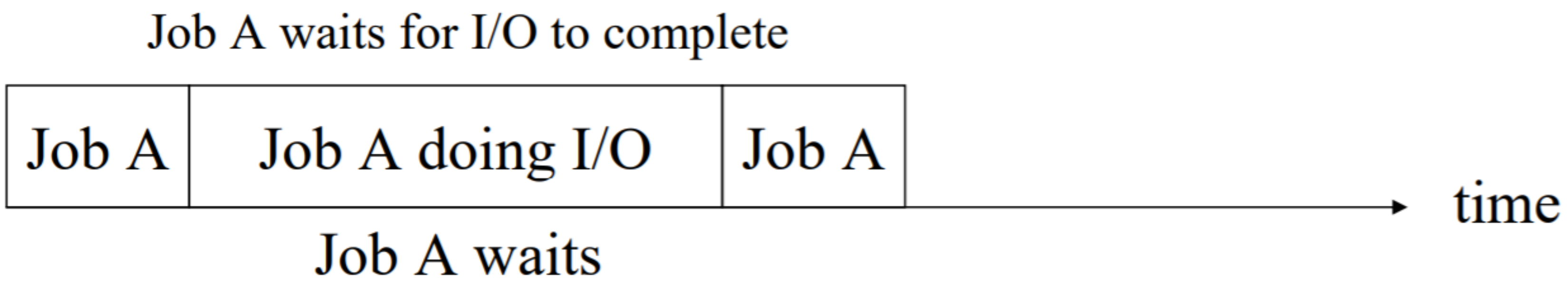
- ➢ Hardware: transistors → ICs
- ➢ Batch processing OSs: collect the jobs (programs and data) together in a batch before processing starts
  - ✓ Automate the sequence of operations.
  - ✓ Introduce Job Control Language (JCL) to describe batch jobs.
  - ✓ Introduce batch monitor.

❑ IBM System/360 (S/360)

- ➢ It used microcode to implement the instruction set, which featured 8-bit byte addressing and binary, decimal and floating-point calculations.

- Batch processing (1960s)

  - Problems: sequential execution; that is, no interaction and no overlap between a fast CPU and slow I/O devices.

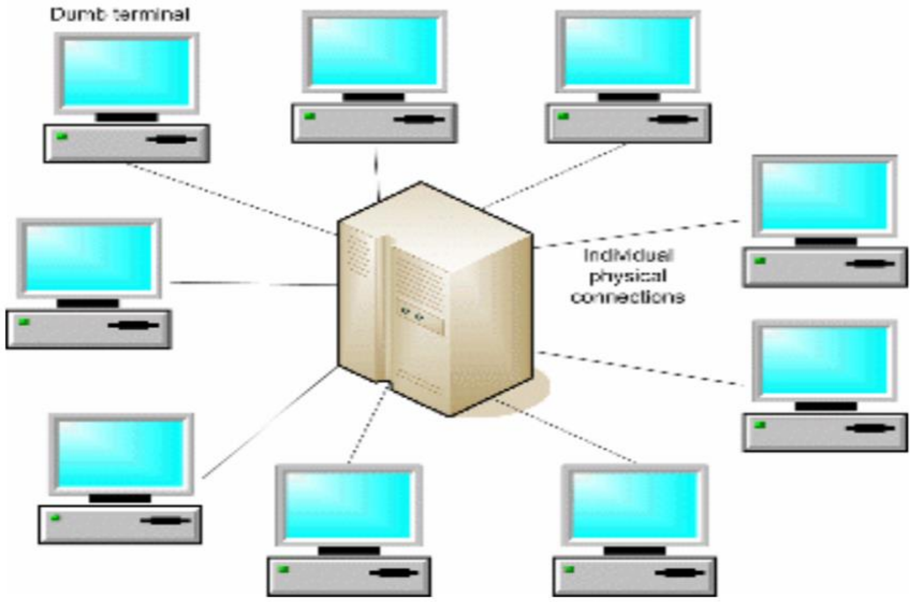


  - Solution:
    - ✓ SPOOL (simultaneous peripheral operation on-line) -- overlap the I/O of a job with other job's computation.

❑ Time Sharing (1960s-1970s)

- ➢ Hardware resources are assigned to different users for a short time period, and thus a user from a terminal gets the feeling that he/she has dedicated hardware resources.

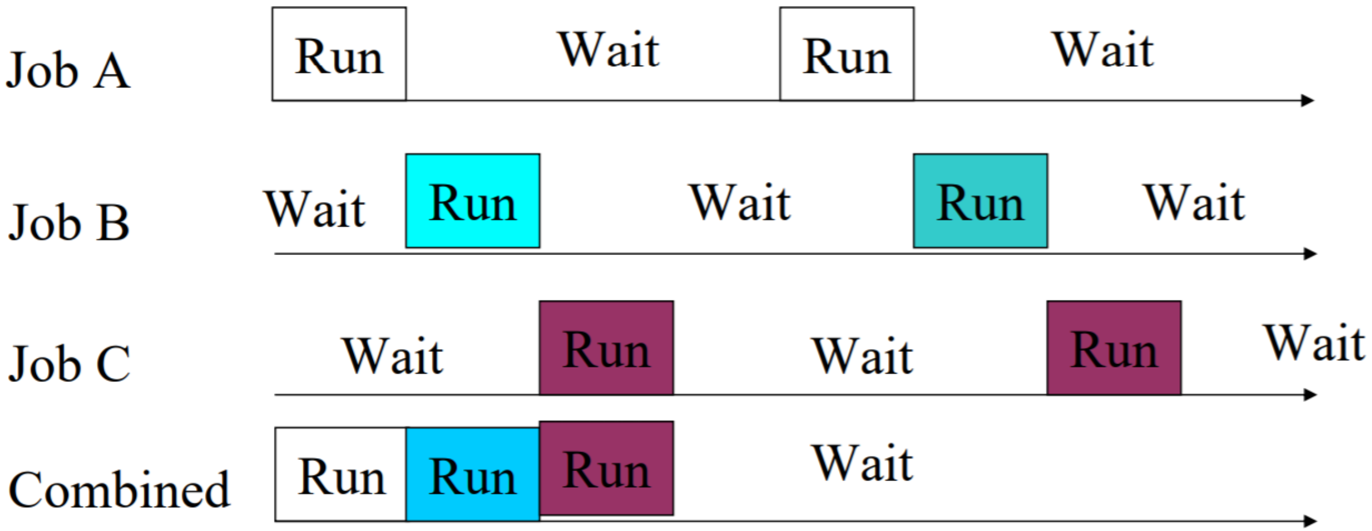
- ➢ CTSS (Compatible Time-Sharing System)--One of the first time-sharing OSs
  - ✓ System clock generates interrupts every 0.2 sec.
  - ✓ At each interrupt, the OS will
    - preempt the current user by swapping the current user's program and data out from the main memory to the hard disk.
    - assign the processor to the next user and swap the next user 's program and data from the hard disk into the main memory.



Time Sharing

❑ Multiprogramming (1970s)

- Having more than one program/jobs in memory at the same time.



Multiprogramming/Multitasking

❑ Multiprogramming (1970s)

➢ Differences between multiprogramming OSs and time sharing OSs.

Multiprogramming	Time Sharing
Allow multiple jobs to share resources	Allow multiple users to share resources
Jobs should be in the memory	Users should be in different terminals, and their jobs may not be in the memory
Goal: to use resource efficiently (i.e., maximize the resource utilization)	Goal: providing a method to fairly share resource among users

Question

- There are three jobs, i.e., Job A, Job B, and Job C, in the queue. The capacity of CPU is 1 MIPS.

	Job A	Job B	Job C
Number of instructions	100 instructions	1000 instructions	500 instructions
I/O time	1100 us	1000 us	700 us
CPU time	$100/(1\times10^6)$ sec=100 us	$1000/(1\times10^6)$ sec=1000 us	$500/(1\times10^6)$ sec=500 us
CPU utilization for running a single job	100/(1100+100)=1/12	1000/(1000+1000)=1/2	500/(500+700)=5/12

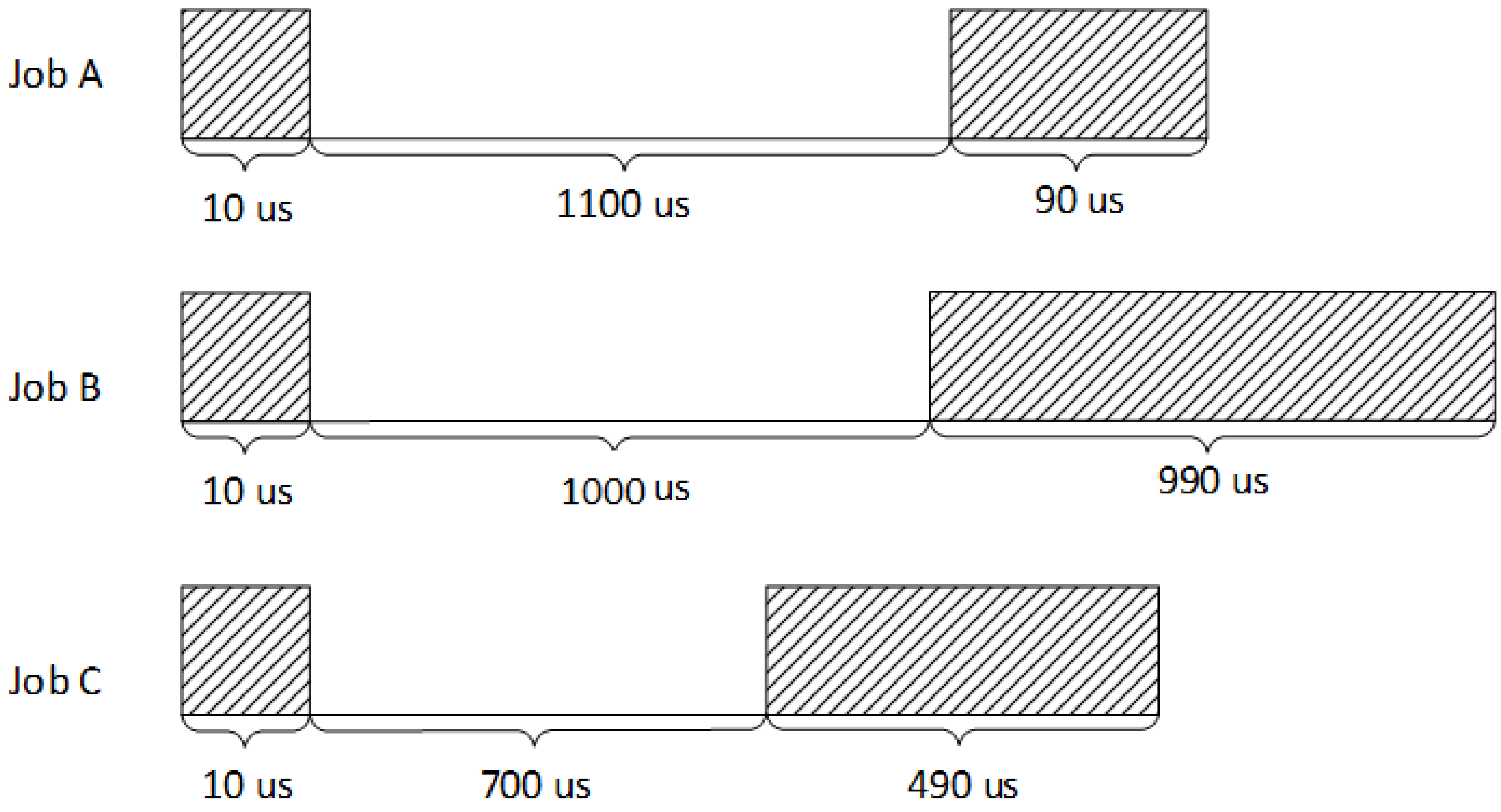
- What is the average CPU utilization for applying simple batching processing and multiprogramming, respectively?

❑ Answer

- Simple Batching Processing
  - ✓ average CPU Utilization =sum(CPU time)/(sum(CPU time)+sum(IO time))
    =4/11

- Multiprogramming
  - ✓ Maximum/Optimal average CPU utilization = min{100%, sum(CPU time of the jobs)/max of the total time among jobs}
    =(100+1000+500)/2000=80%

- One example to explain in which situation the job scheduling is unable to achieve the optimal CPU utilization.



❑ Multiprogramming (1970s)

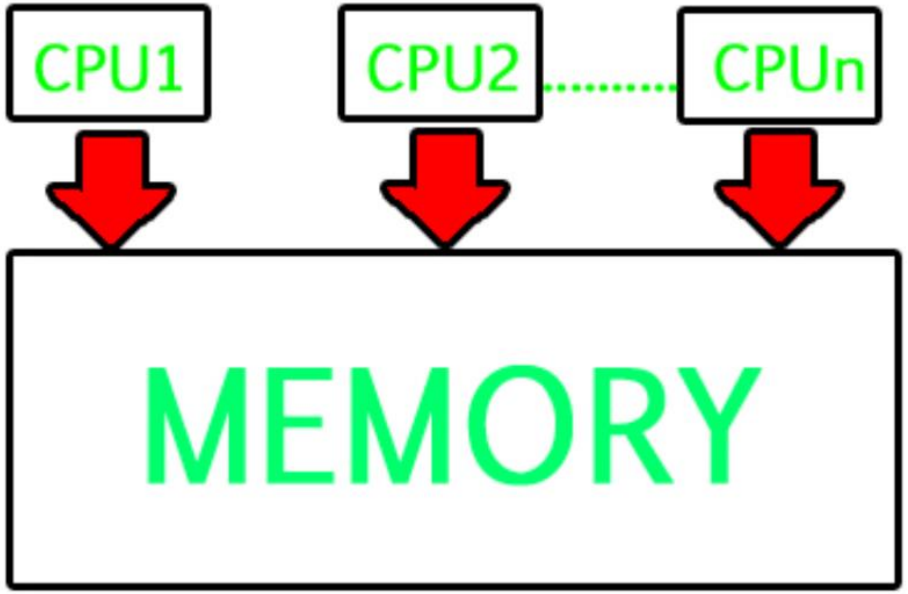
- ➢ Typical machines: IBM 360/370, PDP-7/11, Intel 8080
- ➢ Build the foundation of OS
  - ✓ The decision on which job to execute next from a queue/pool of ready jobs involves CPU scheduling.
  - ✓ Having several jobs ready to run implies that they must reside in memory, which requires Memory management.
  - ✓ Jobs may have to be swapped in/out of main memory to the disk implies that Disk management must be provided.
  - ✓ Multiple jobs running implies that OS must minimize the impact of one job on another, which introduces Protection.
- ➢ Birth of UNIX

❑ Multiprocessing

- Different processes can be assigned to different processors (cores) for their execution.

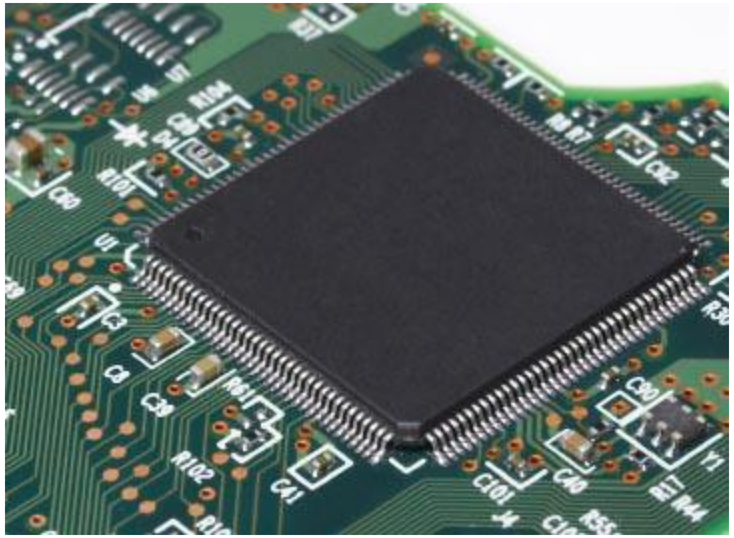
- Multiprocessing refers to the hardware (i.e., the CPU units) rather than the software (i.e., operating systems).

- A system can be both multiprogramming by having multiple programs running at the same time and multiprocessing by having more than one physical processors.



OS/Computer Evolution

- Personal computers (1980s)
  - Hardware: ICs → LSI → VLSI →ULSI
  - Introduce microcomputers
  - Introduce GUI (Graphical User Interface) for OSs
  - Birth of MS-DOS, and then windows…
  - Typical machines: Intel 80286/80386/…, IBM PC, Macintosh



OS/Computer Evolution

- Summary

  - 1st generation OS--Serial Processing OS
    - ✓ Manual operations on a bare machine
  - 2nd generation OS—Batch Processing OS
    - ✓ Introduce Job Control Language (JCL) to instruct the system on how to automatically run batch jobs
  - 3rd generation OS—Time Sharing OS and Multiprogramming OS
    - ✓ Sharing resources among jobs/users
  - 4th generation OS —OS on PCs
    - ✓ Introduce GUI

What's Next?

❑ Quantum Computer

- A quantum computer is a type of computer that uses quantum mechanics so that it can perform certain kinds of computation more efficiently than a classical computer can.

- Difference between classical computer and quantum computer

  - ✓ In a classical computer, information (such as numbers, text, and images) can be represented by a number of bits, where a bit can be set to either 0 or 1.
  - ✓ A quantum computer does not use bits to store information. Instead, it uses qubits. Each qubit can not only be set to 1 or 0, but it can also be set to 1 and 0.