Introduction to Operating Systems (Part I)

Amir H. Payberah
amir@sics.se

Amirkabir University of Technology
(Tehran Polytechnic)

Course Information

Course Objective

- The purpose of this course is to teach the design of operating systems.

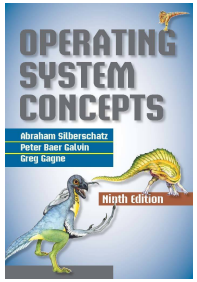
Course Objective

- The purpose of this course is to teach the design of operating systems.

- Topics we will cover include:
  - Process management
  - Memory management
  - File systems
  - I/O management
  - Security and privacy

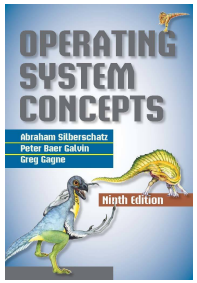
Course Textbooks

- Operating System Concepts, 9th Edition
  Avil Silberschatz et al., Wiley, 2013

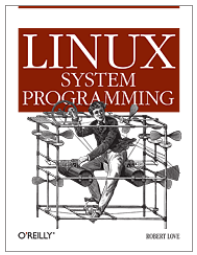


Course Textbooks

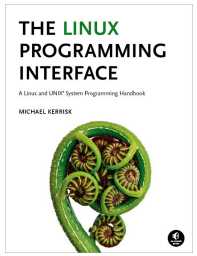
- Operating System Concepts, 9th Edition
  Avil Silberschatz et al., Wiley, 2013



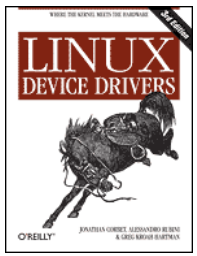
- Linux System Programming, 2nd Edition
  Robert Love, O'Relly Media, 2013



- The Linux Programming Interface
  Michael Kerrisk, No Starch Press, 2010



- Linux Device Drivers, 3rd Edition
  Jonathan Corbet et al., O'Relly Media, 2005



Course Examination

- Mid term exam: 30%
- Final exam: 30%
- Lab assignments: 40%
  - Seven programming assignments in C
  - Students will work in groups of three

Course Web Site

- You can find all the course information on the course web page: http://www.sics.se/~amir/os14.htm
- Use the course discussion forum if you have any questions.

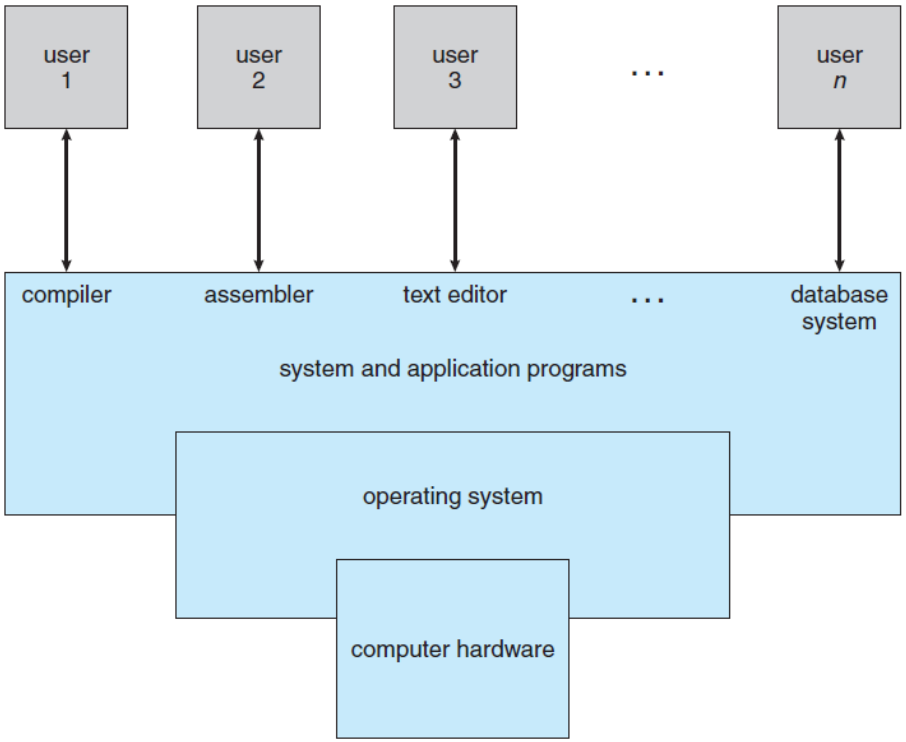
What is an Operating System?

What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.

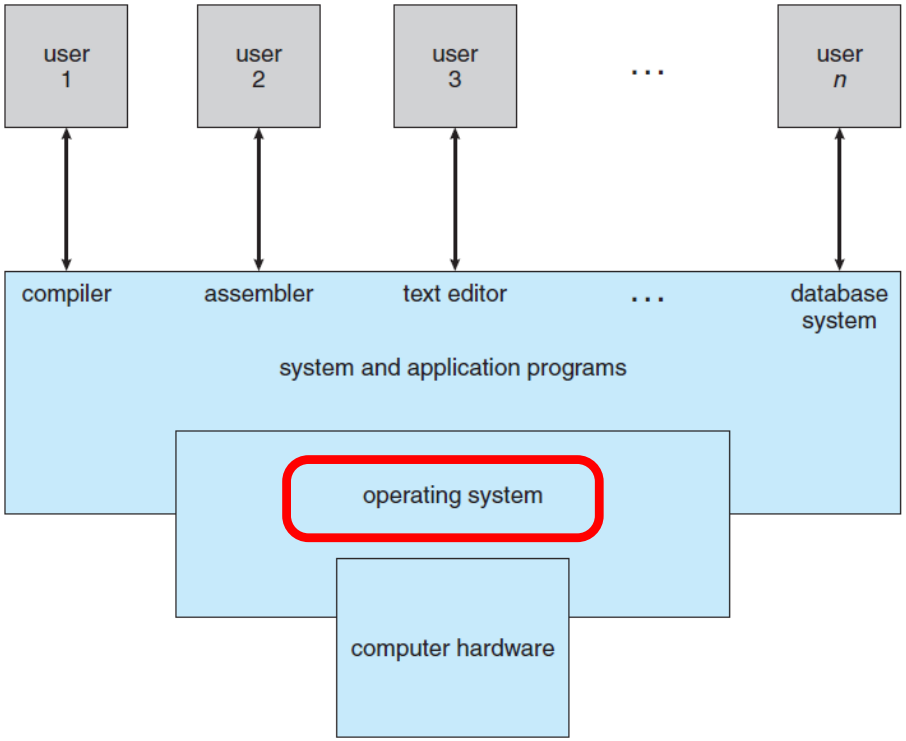
What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.



What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.



Operating System Goals

- Execute user programs and make solving user problems easier.

Operating System Goals

- Execute user programs and make solving user problems easier.
- Make the computer system convenient to use.

Operating System Goals

- Execute user programs and make solving user problems easier.
- Make the computer system convenient to use.
- Use the computer hardware in an efficient manner.

What Operating Systems Do

- OS is a resource allocator
  - Manages all resources.
  - Decides between conflicting requests for efficient and fair resource use.

What Operating Systems Do

- OS is a resource allocator
  - Manages all resources.
  - Decides between conflicting requests for efficient and fair resource use.

- OS is a program controller
  - Controls execution of programs to prevent errors and improper use of the computer.

Operating Systems Definition

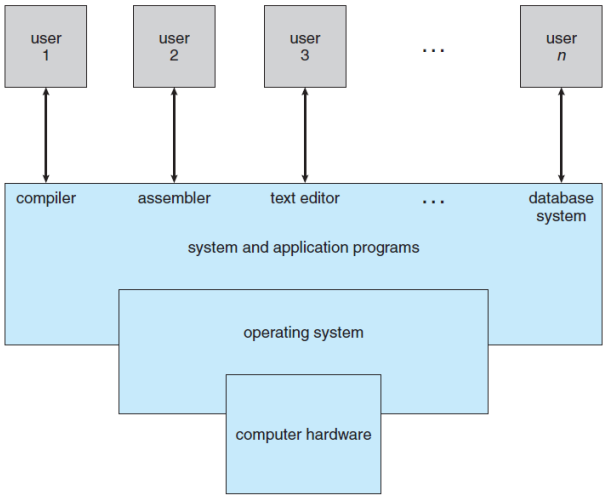
- No universally accepted definition.

Operating Systems Definition

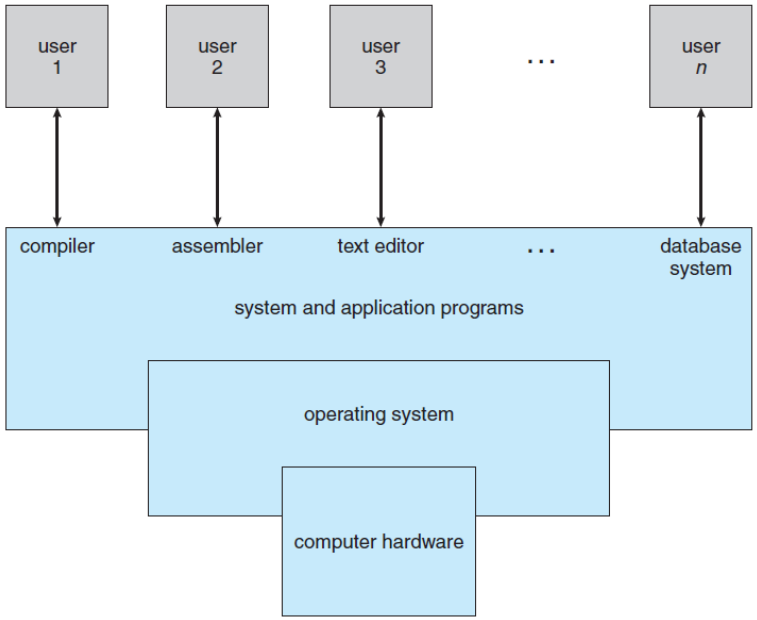
- No universally accepted definition.
- The operating system is the one program running at all times on the computer, usually called the kernel.

Operating Systems Definition

- No universally accepted definition.
- The operating system is the one program running at all times on the computer, usually called the kernel.
- Everything else is either a system program or an application program.

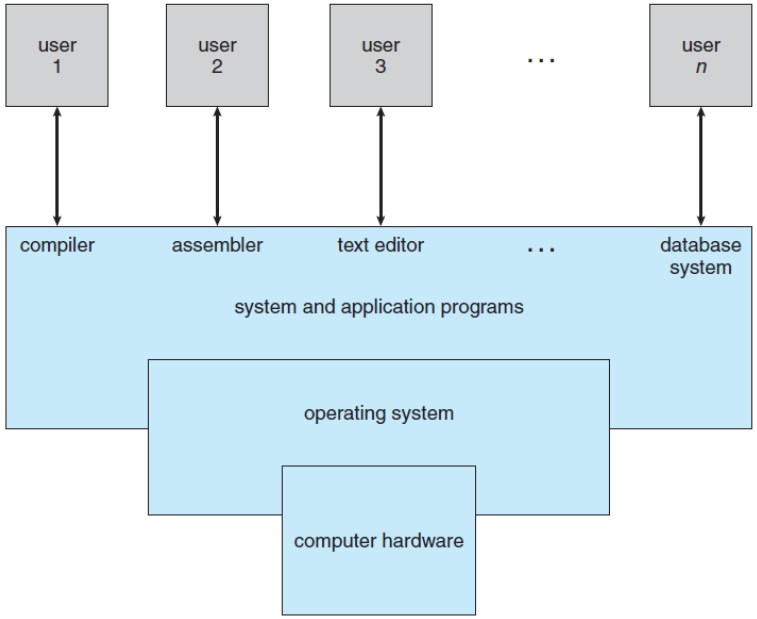


Four Components of a Computer System (1/4)



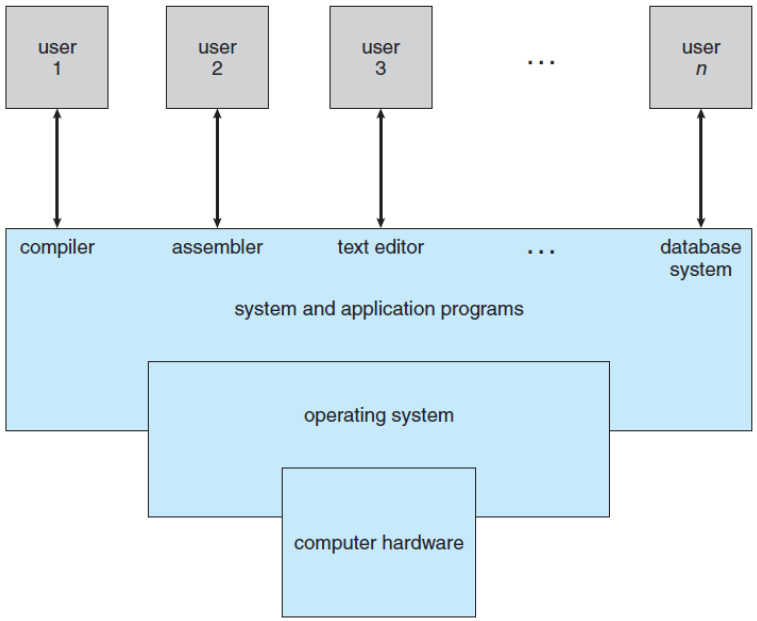
- Hardware
  - Provides basic computing resources.
  - CPU, memory, I/O devices, ...

Four Components of a Computer System (2/4)



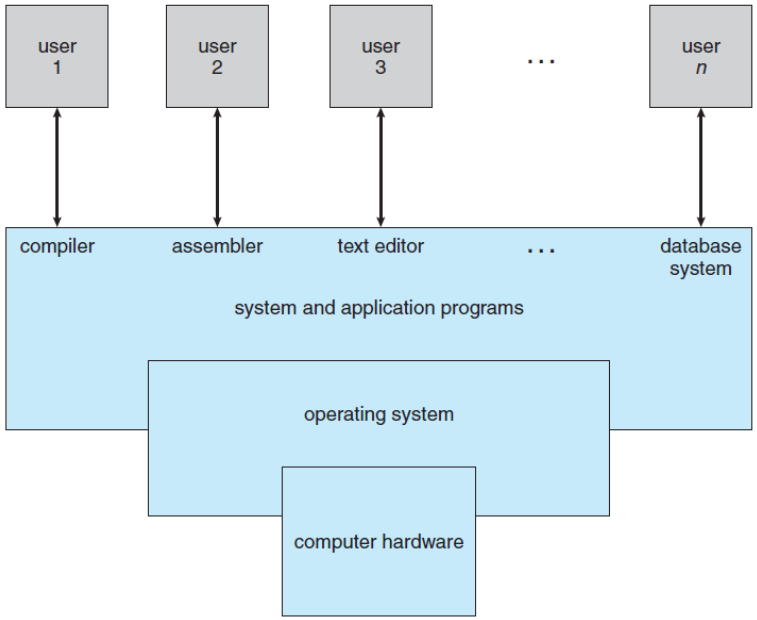
- Operating system
  - Controls and coordinates use of hardware among various applications and users.

Four Components of a Computer System (3/4)



- Application programs
  - Compilers, web browsers, database systems, video games, ...
- System programs
  - File manipulation, program loading and execution, ...

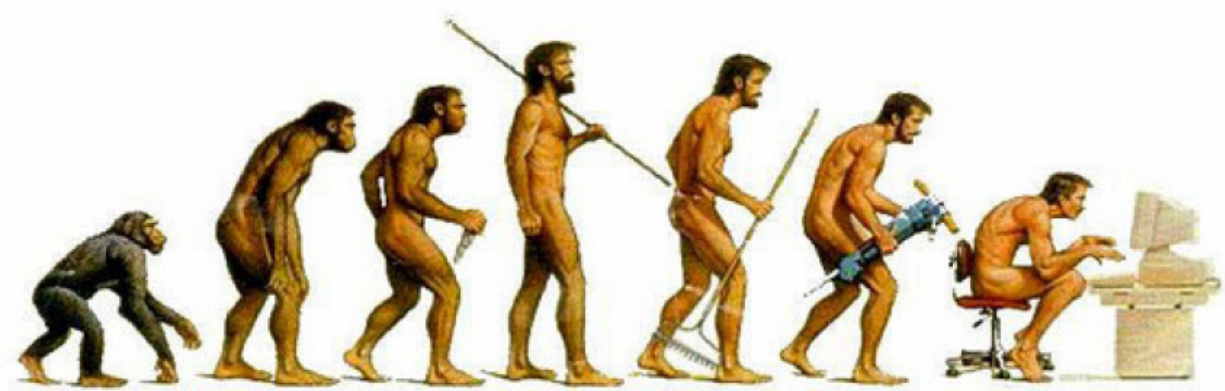
Four Components of a Computer System (4/4)



- Users
  - People, machines, other computers

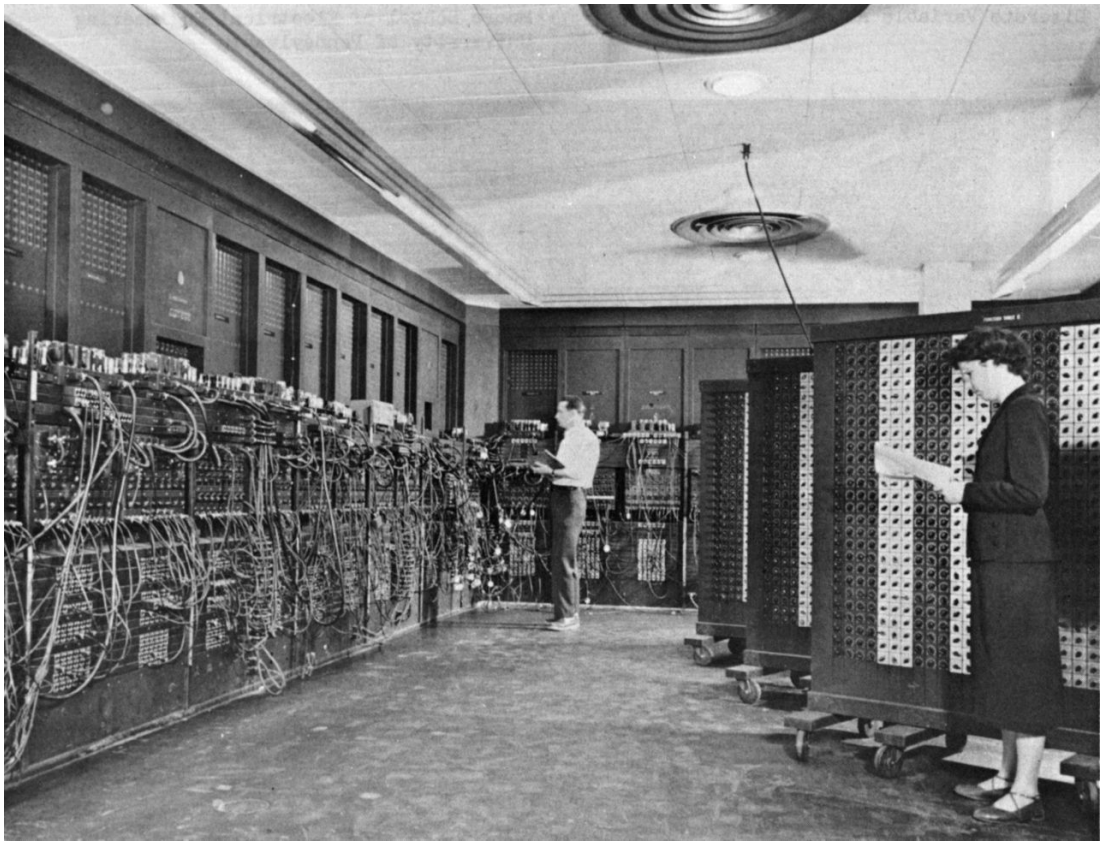
Operating Systems

A Brief History of Operating Systems



First Generation: 1945-1955 (1/3)

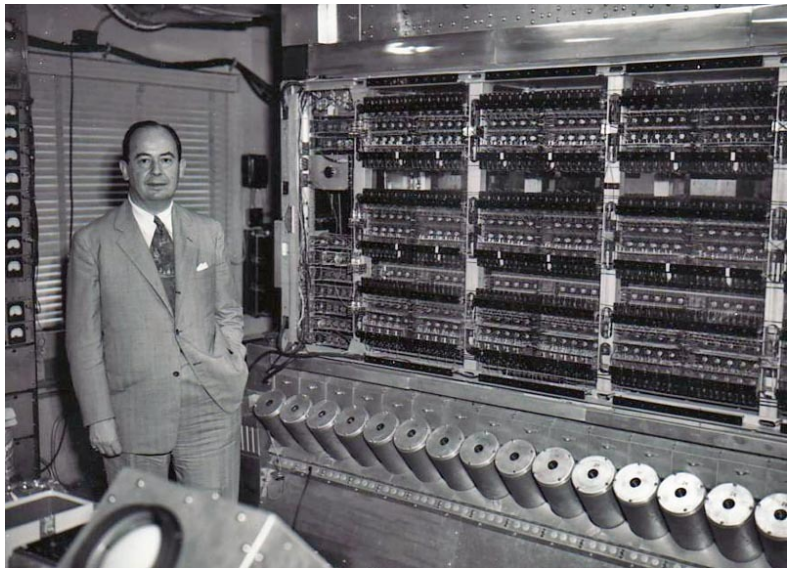
- No operating system
- Vacuum tubes and plugboards



ENIAC (Electronic Numerical Integrator And Computer): the first electronic general-purpose computer.
[http://en.wikipedia.org/wiki/ENIAC]

First Generation: 1945-1955 (2/3)

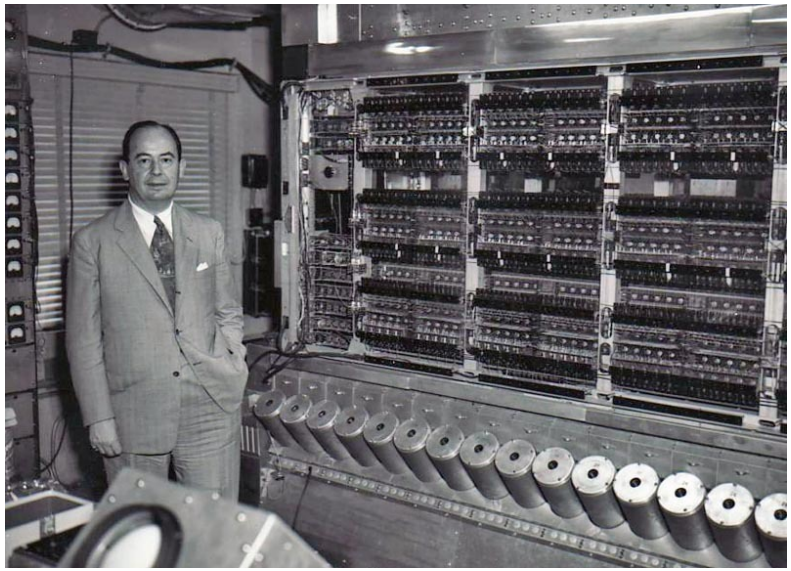
- Human was the operator and programmer.



John von Neumann
[http://ysfine.com/wigner/neumann.html]

First Generation: 1945-1955 (2/3)

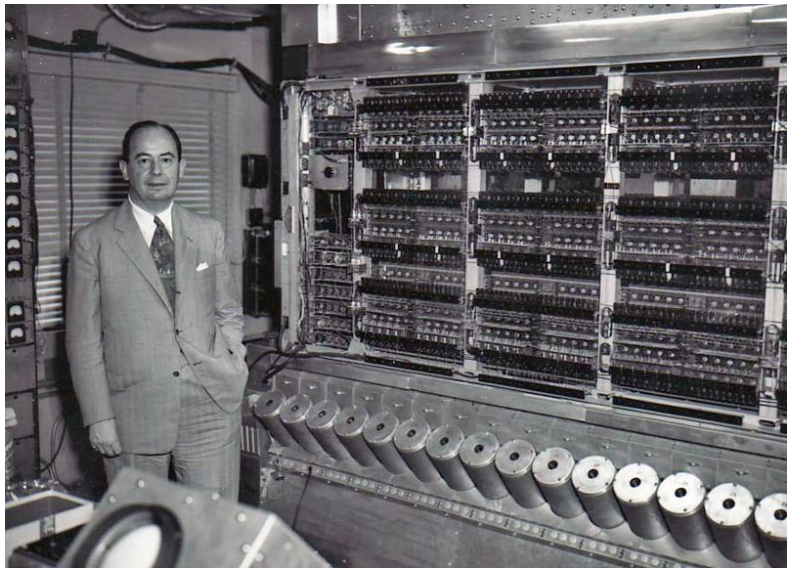
- Human was the operator and programmer.
- Computer were programmed by physically re-wiring it; later, through stored programs (von Neumann architecture).



John von Neumann
[http://ysfine.com/wigner/neumann.html]

First Generation: 1945-1955 (2/3)

- Human was the operator and programmer.
- Computer were programmed by physically re-wiring it; later, through stored programs (von Neumann architecture).
- Programs written in machine or assembly language.



John von Neumann
[http://ysfine.com/wigner/neumann.html]

First Generation: 1945-1955 (3/3)

- Problems:
  - Serial processing: users had access to the computer one by one in series.
  - Users have to write again and again the same routines.

Second Generation: 1955-1965 (1/5)

- Transistors
- Mainframes

IBM 7094 at Columbia University
[http://www.columbia.edu/cu/computinghistory/1965.html]

Second Generation: 1955-1965 (2/5)

- Separation between operators and programmers.
  - The programmer: prepares her/his job off-line.
  - The operator: runs the job and delivers a printed output.

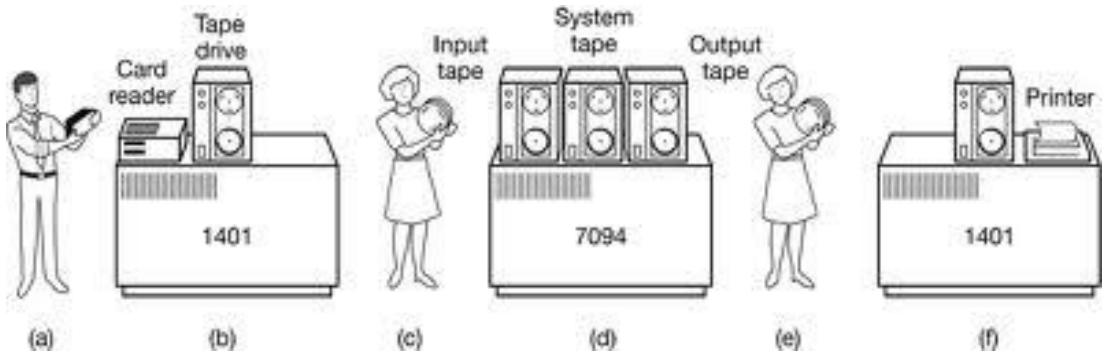
Second Generation: 1955-1965 (2/5)

- Separation between operators and programmers.
  - The programmer: prepares her/his job off-line.
  - The operator: runs the job and delivers a printed output.

- Job
  - A program or set of programs
  - A programmer would first write the program on paper (in FORTRAN or assembly), then punch it on cards.

Second Generation: 1955-1965 (3/5)

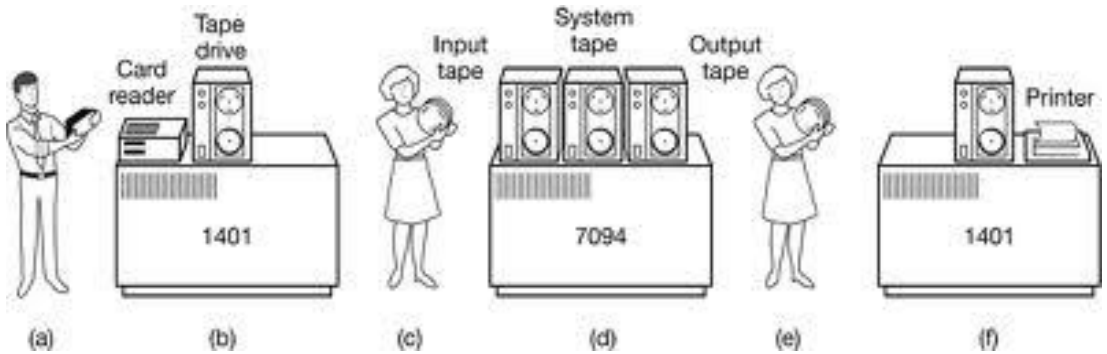
- Batch the jobs together.



[A.S. Tanenbaum et al., Operating Systems Design and Implementation, 2006]

Second Generation: 1955-1965 (3/5)

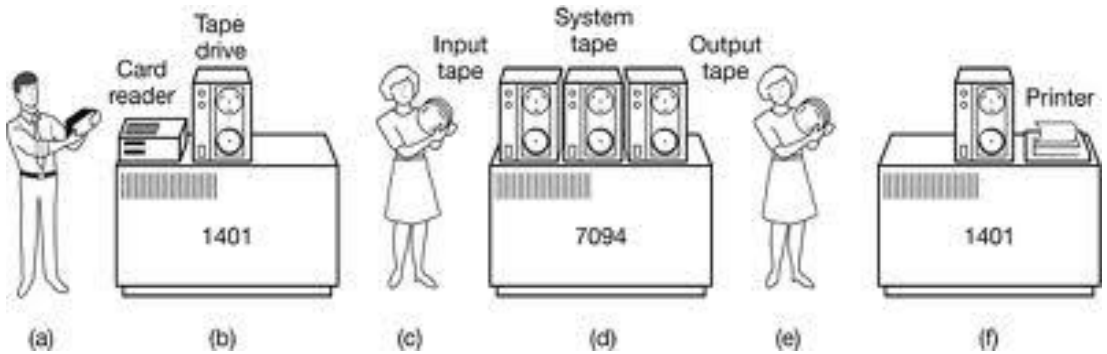
- Batch the jobs together.
- The operator pre-reads jobs onto a magnetic tape.



[A.S. Tanenbaum et al., Operating Systems Design and Implementation, 2006]

Second Generation: 1955-1965 (3/5)

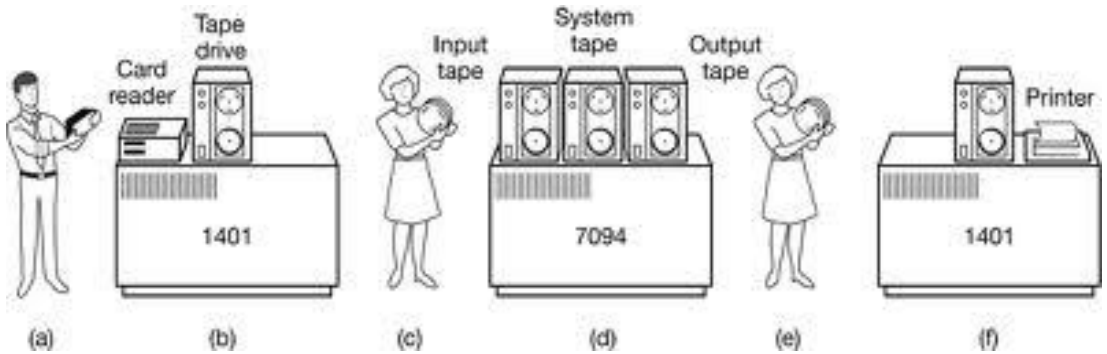
- Batch the jobs together.
- The operator pre-reads jobs onto a magnetic tape.
- The operator loads a special program (monitor) that reads the jobs from the tape and run them sequentially.



[A.S. Tanenbaum et al., Operating Systems Design and Implementation, 2006]

Second Generation: 1955-1965 (3/5)

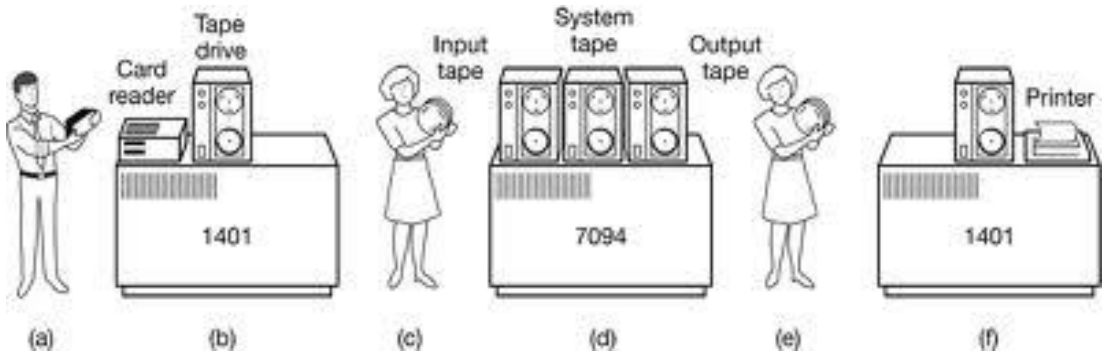
- Batch the jobs together.
- The operator pre-reads jobs onto a magnetic tape.
- The operator loads a special program (monitor) that reads the jobs from the tape and run them sequentially.
- The monitor program writes the output of each job on a second magnetic tape.



[A.S. Tanenbaum et al., Operating Systems Design and Implementation, 2006]

Second Generation: 1955-1965 (3/5)

- Batch the jobs together.
- The operator pre-reads jobs onto a magnetic tape.
- The operator loads a special program (monitor) that reads the jobs from the tape and run them sequentially.
- The monitor program writes the output of each job on a second magnetic tape.
- The operator brings the full output tape for offline printing.



[A.S. Tanenbaum et al., Operating Systems Design and Implementation, 2006]

Second Generation: 1955-1965 (4/5)

- Monitor
  - It is the ancestor of modern O/S.

Second Generation: 1955-1965 (4/5)

- Monitor
  - It is the ancestor of modern O/S.
  - A special program that controls the sequence of events.

Second Generation: 1955-1965 (4/5)

- Monitor
  - It is the ancestor of modern O/S.
  - A special program that controls the sequence of events.
  - Always resides in main memory.

Second Generation: 1955-1965 (4/5)

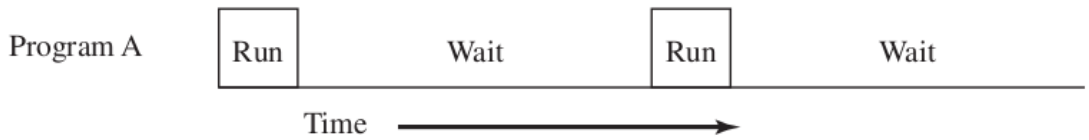
- Monitor
  - It is the ancestor of modern O/S.
  - A special program that controls the sequence of events.
  - Always resides in main memory.
  - Reads in jobs one at a time, places a job in the user program area of the memory, and passes control to it.

Second Generation: 1955-1965 (4/5)

- Monitor
  - It is the ancestor of modern O/S.
  - A special program that controls the sequence of events.
  - Always resides in main memory.
  - Reads in jobs one at a time, places a job in the user program area of the memory, and passes control to it.
  - Upon completion, the user program branches back to the monitor, which immediately loads and executes the next job

Second Generation: 1955-1965 (5/5)

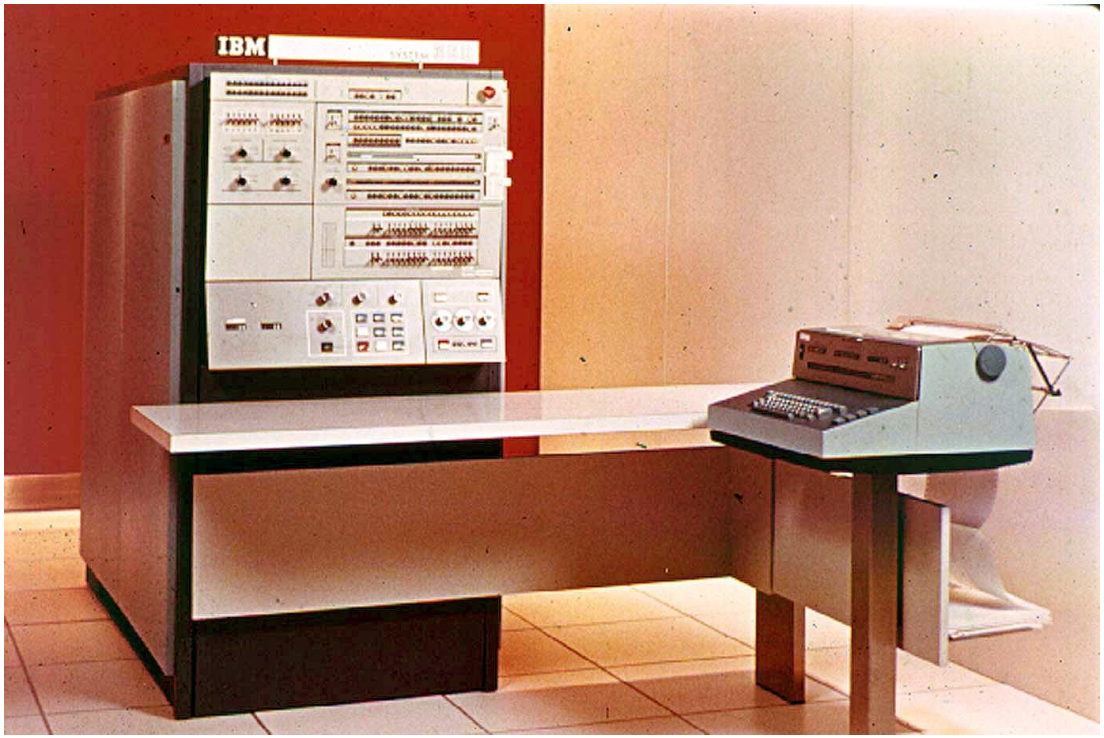
- Problems:
  - A lot of CPU time is still wasted waiting for I/O instructions to complete.
  - I/O devices much slower than processor.



[W. Stallings, Operating Systems: Internals and Design Principles, 2011]

Third Generation: 1965-1980 (1/5)

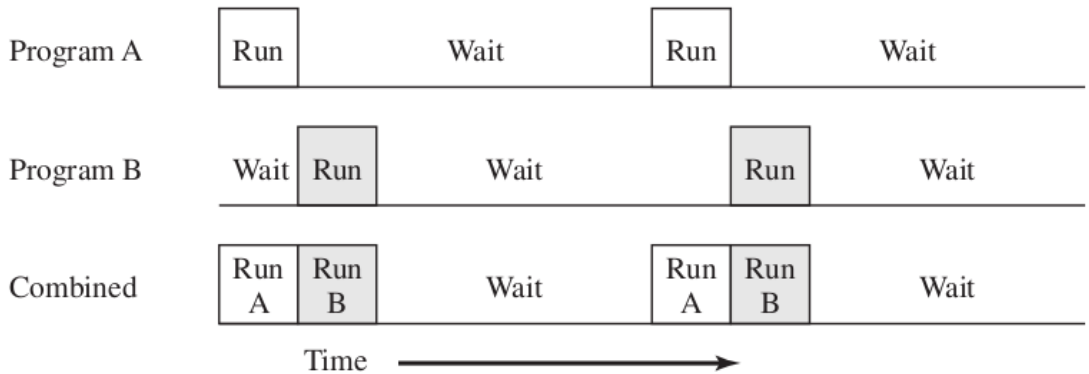
- Integrated Circuits (ICs)



IBM 360
[http://www.computermuseum.li/Testpage/IBM-360-1964.htm]

Third Generation: 1965-1980 (2/5)

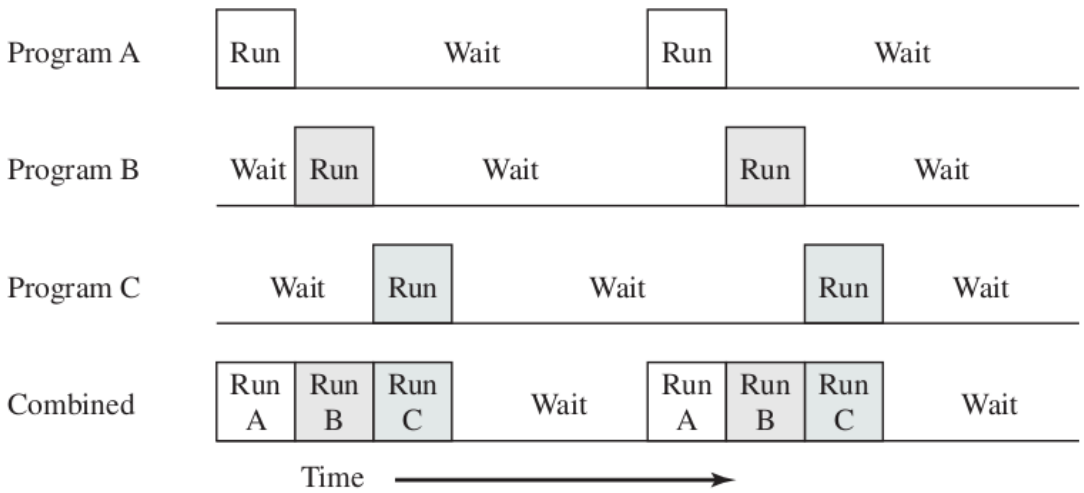
- Multiprogrammed batch systems.
- Load two jobs in memory: while one job is waiting for I/O, the processor could switch to the other job.



[W. Stallings, Operating Systems: Internals and Design Principles, 2011]

Third Generation: 1965-1980 (3/5)

- Expand to three, four or more jobs.
- Jobs are kept in main memory at the same time and the CPU is multiplexed among them or multiprogrammed.



[W. Stallings, Operating Systems: Internals and Design Principles, 2011]

Third Generation: 1965-1980 (4/5)

- Multiprogramming alone does not give any guarantee that a program will run in a timely manner.

Third Generation: 1965-1980 (4/5)

- Multiprogramming alone does not give any guarantee that a program will run in a timely manner.
- Tasks kept running until they performed an operation that required waiting for an external event such as I/O.

Third Generation: 1965-1980 (4/5)

- Multiprogramming alone does not give any guarantee that a program will run in a timely manner.
- Tasks kept running until they performed an operation that required waiting for an external event such as I/O.
- But, in a multiple-user system, users want to see their program running as if it was the only program in the computer.

Third Generation: 1965-1980 (4/5)

- Multiprogramming alone does not give any guarantee that a program will run in a timely manner.
- Tasks kept running until they performed an operation that required waiting for an external event such as I/O.
- But, in a multiple-user system, users want to see their program running as if it was the only program in the computer.
- Solution? time-sharing or preemptive multitasking systems.

Third Generation: 1965-1980 (5/5)

- Time-sharing
  - A logical extension of multiprogramming for handling multiple interactive jobs among multiple users.
  - Hardware timer interrupt: switching jobs.

Third Generation: 1965-1980 (5/5)

- Time-sharing
  - A logical extension of multiprogramming for handling multiple interactive jobs among multiple users.
  - Hardware timer interrupt: switching jobs.

- Birth of UNIX: CTSS $\rightarrow$ MULTICS $\rightarrow$ UNIX

UNIX®

Fourth Generation: 1980-Present (1/3)

- Personal Computers (PCs)

Fourth Generation: 1980-Present (2/3)

- From multiple users back to a single user.
- Multitasking a central feature of modern PC operating systems.
- PC systems emphasize user convenience.

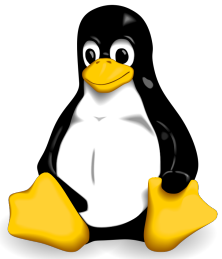
Fourth Generation: 1980-Present (3/3)

- GNU (GNU's Not Unix!): 1983

- Mac OS: 1984

- Microsoft Windows: 1985

- Linux: 1991



GNU/Linux (1/4)

- In 1971 Richard Matthew Stallman (RMS) joined MIT.

GNU/Linux (1/4)

- In 1971 Richard Matthew Stallman (RMS) joined MIT.
- At that time, all the programmers used to share their code freely.

GNU/Linux (1/4)

- In 1971 Richard Matthew Stallman (RMS) joined MIT.
- At that time, all the programmers used to share their code freely.
- In 1980, software companies refused to share the code (copyright).

GNU/Linux (1/4)

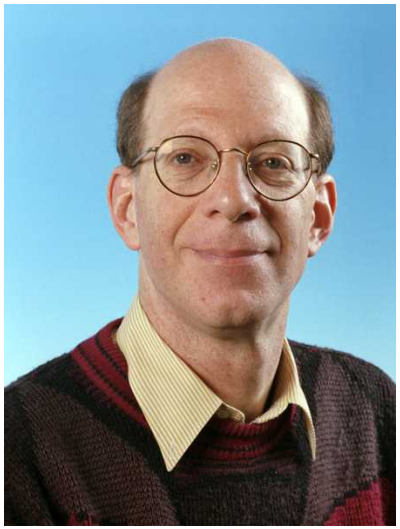
- In 1971 Richard Matthew Stallman (RMS) joined MIT.
- At that time, all the programmers used to share their code freely.
- In 1980, software companies refused to share the code (copyright).
- In 1985, in response, Stallman, founded the Free Software Foundation (FSF) and published the GNU manifesto.
  - Outlined his motivation for creating a free OS (GNU), which would be compatible with Unix.

GNU/Linux (2/4)

- In 1989, Stallman released the first program independent GNU General Public License (GPL) or copyleft.
- Now the only thing that GNU lacked was a completely free OS kernel: GNU Hurd kernel

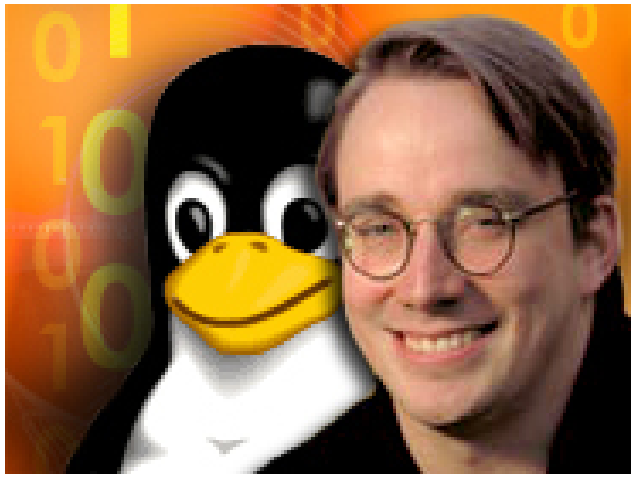
GNU/Linux (3/4)

- In 1985, Andy Tanenbaum wrote a Unix like OS from scratch, called Minix.
- He implemented it for educational purposes.



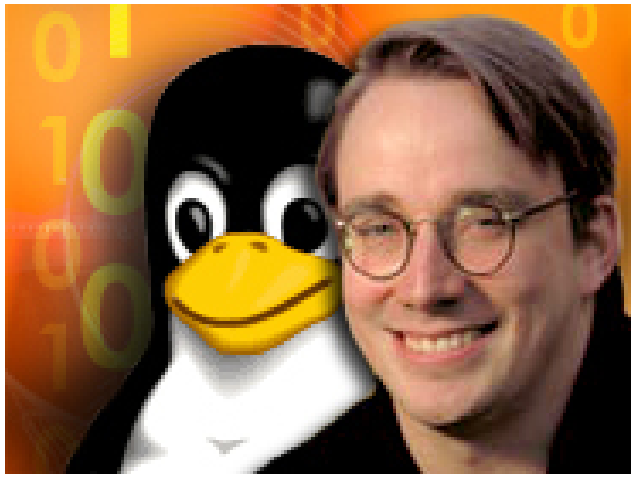
GNU/Linux (4/4)

- In 1990, Linus Torvalds wanted to improve Minix.



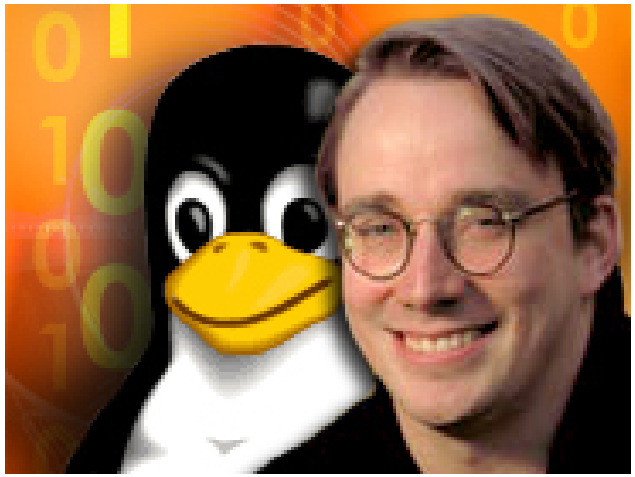
GNU/Linux (4/4)

- In 1990, Linus Torvalds wanted to improve Minix.
- But he was prohibited by Tanenbaum to do so.



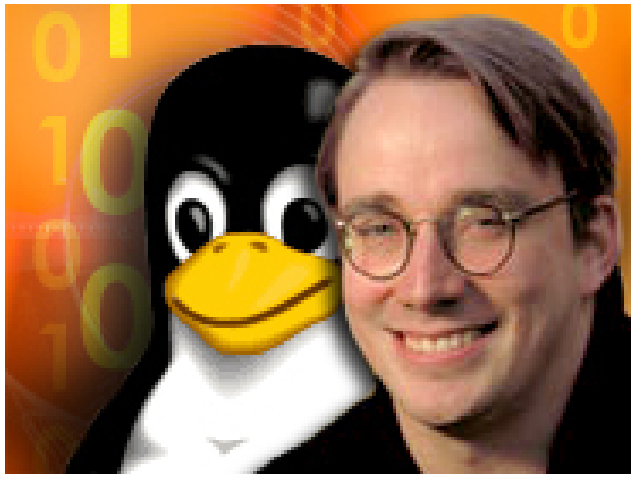
GNU/Linux (4/4)

- In 1990, Linus Torvalds wanted to improve Minix.
- But he was prohibited by Tanenbaum to do so.
- So, Linus implemented his own kernel and released it under GPL: Linux kernel



GNU/Linux (4/4)

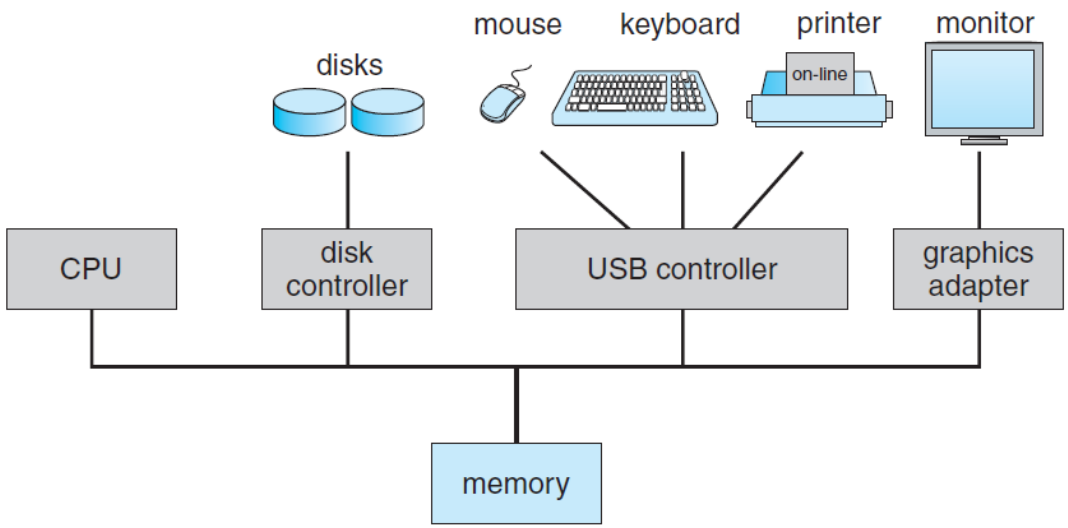
- In 1990, Linus Torvalds wanted to improve Minix.
- But he was prohibited by Tanenbaum to do so.
- So, Linus implemented his own kernel and released it under GPL: Linux kernel
- Linux, is then, used as the kernel of the GNU in many distributions.



Computer System Organization

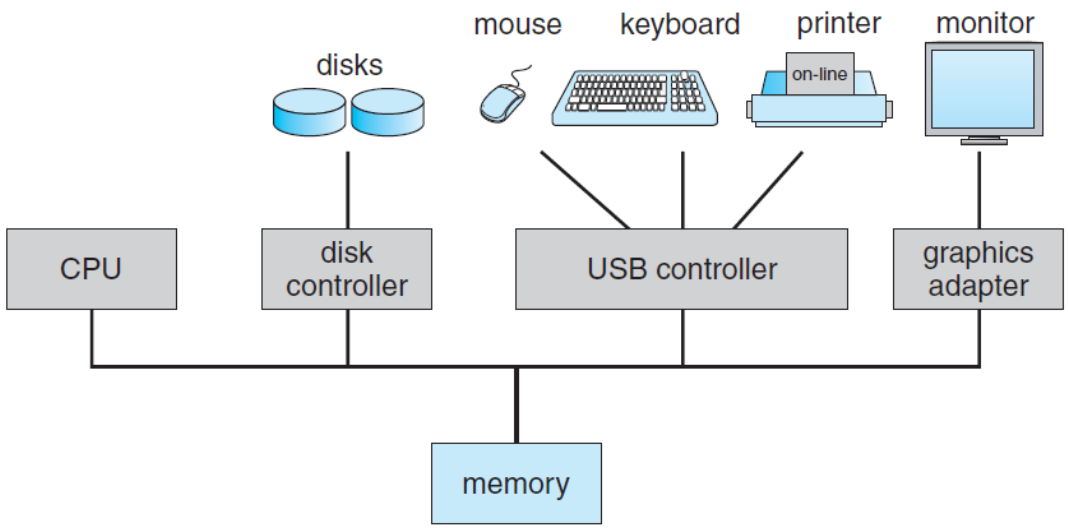
Computer-System Operation

- One or more CPUs, and device controllers connect through common bus providing access to shared memory.



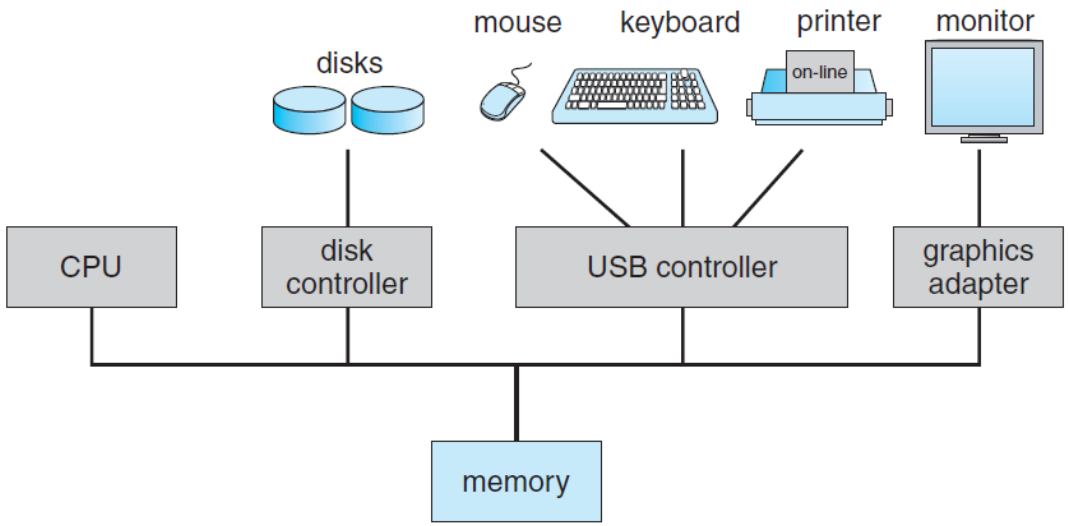
Computer-System Operation

- One or more CPUs, and device controllers connect through common bus providing access to shared memory.
- The CPU and the device controllers can execute in parallel, competing for memory cycles.



Computer-System Operation

- One or more CPUs, and device controllers connect through common bus providing access to shared memory.
- The CPU and the device controllers can execute in parallel, competing for memory cycles.
- Device controllers inform CPU that it is finished with the operation by causing an interrupt.



Interrupt (1/3)

- Hardware may trigger an interrupt at any time by sending a signal to the CPU.

Interrupt (1/3)

- Hardware may trigger an interrupt at any time by sending a signal to the CPU.
- Software may trigger an interrupt by executing a special operation called a system call.

Interrupt (2/3)

- When the CPU is interrupted, it stops what it is doing and immediately transfers execution to an address where the service routine for the interrupt is located.

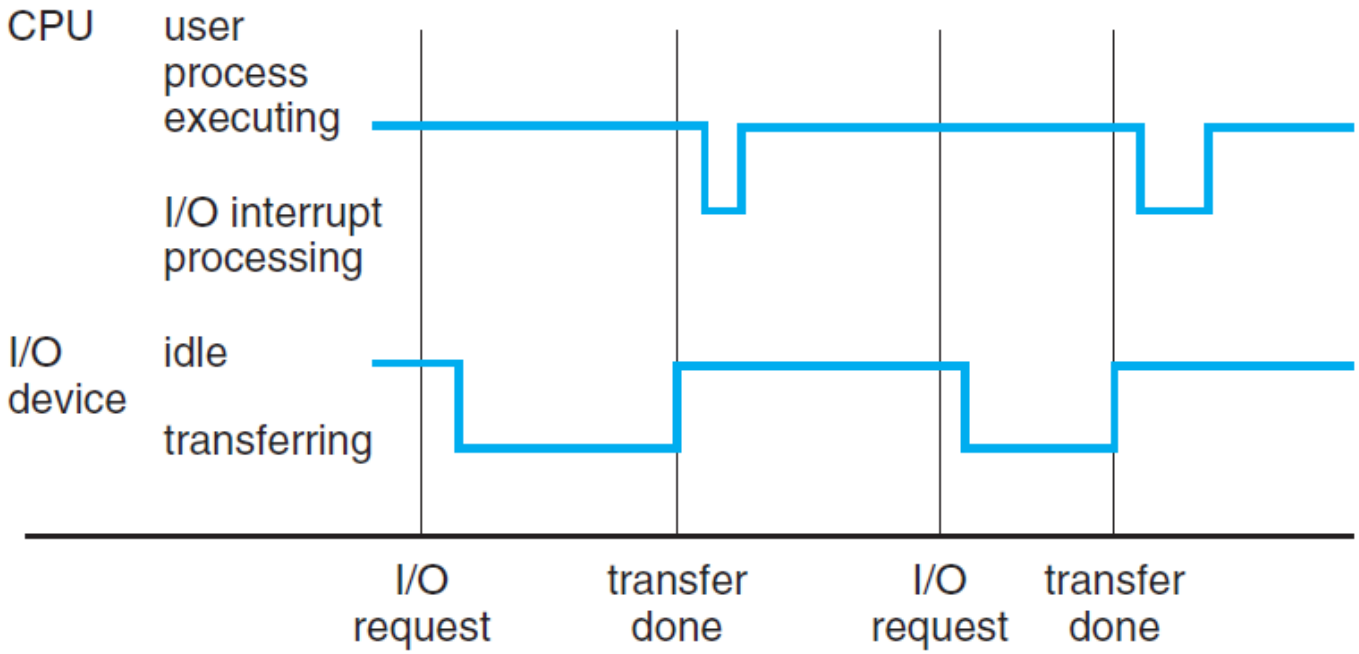
Interrupt (2/3)

- When the CPU is interrupted, it stops what it is doing and immediately transfers execution to an address where the service routine for the interrupt is located.
- The CPU resumes the interrupted computation, when the interrupt service routine completes.

Interrupt (2/3)

- When the CPU is interrupted, it stops what it is doing and immediately transfers execution to an address where the service routine for the interrupt is located.
- The CPU resumes the interrupted computation, when the interrupt service routine completes.
- The OS preserves the state of the CPU by storing registers and the program counter.

Interrupt (3/3)



Storage Structure (1/2)

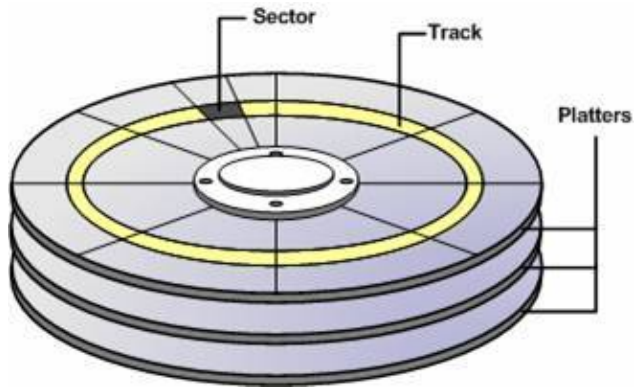
- Main memory (RAM)
  - The CPU can load instructions only from memory.
  - Typically volatile

Storage Structure (1/2)

- Main memory (RAM)
  - The CPU can load instructions only from memory.
  - Typically volatile
- Secondary storage
  - Extension of main memory that provides large nonvolatile storage capacity.
  - E.g., magnetic disk and SSD

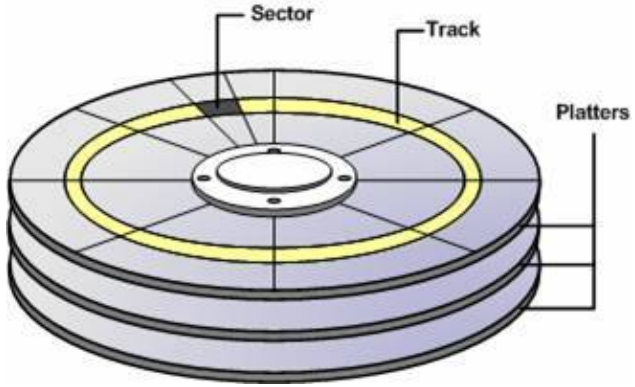
Storage Structure (2/2)

- Magnetic disks
  - Disk surface is logically divided into tracks, which are subdivided into sectors.



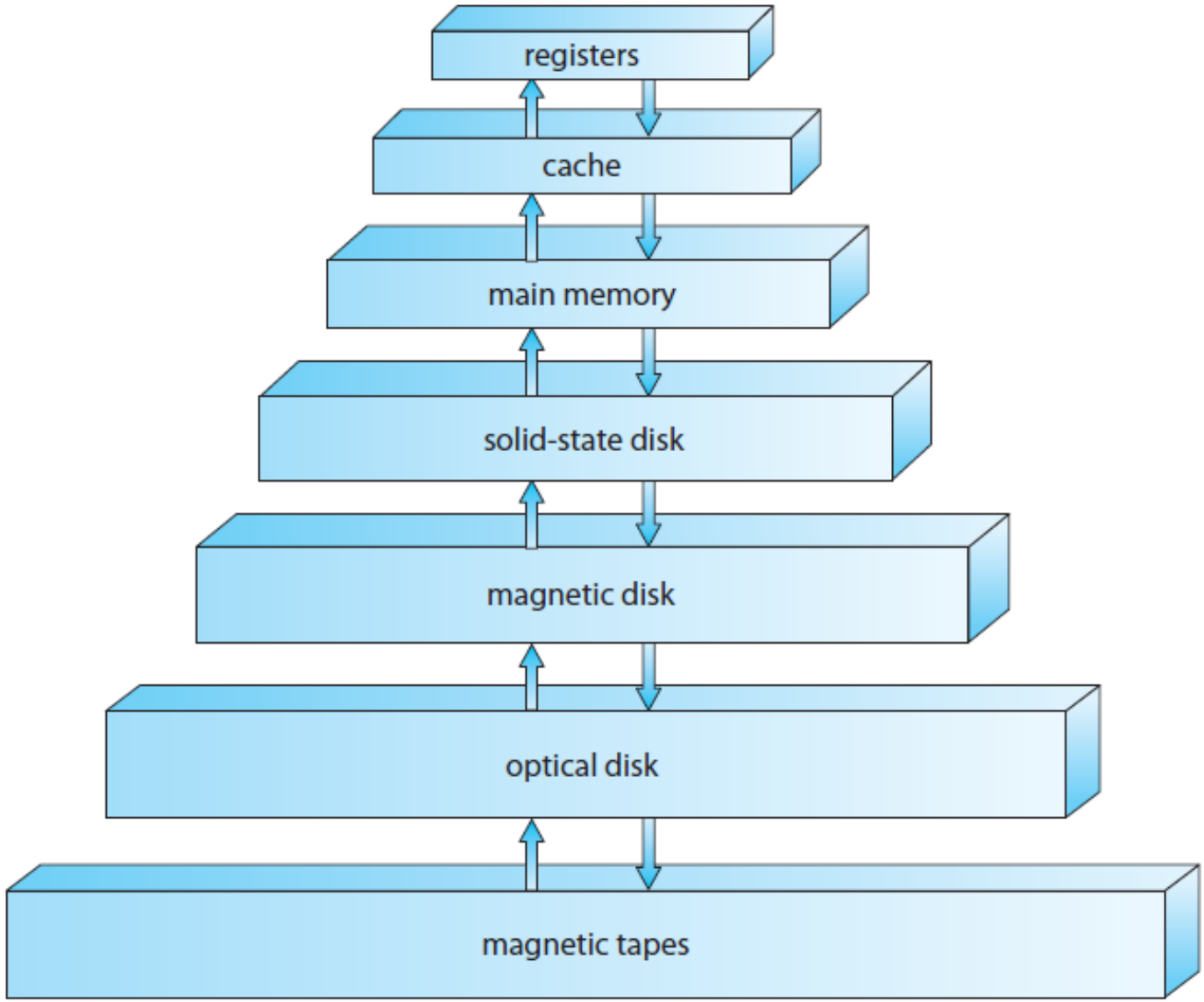
Storage Structure (2/2)

- Magnetic disks
  - Disk surface is logically divided into tracks, which are subdivided into sectors.



- Solid-state disks (SSD)
  - Faster than hard disks, nonvolatile.
  - Becoming more popular.

Storage Hierarchy - Cost vs. Speed



Summary

Summary

- OS history
  - First generation: no OS
  - Second generation: mainframes, batch programming
  - Third generation: multiprogramming, multitasking
  - Fourth generation: PCs

Summary

- OS history
  - First generation: no OS
  - Second generation: mainframes, batch programming
  - Third generation: multiprogramming, multitasking
  - Fourth generation: PCs
- Computer-system organization
  - I/O devices
  - Interrupt
  - Storage

Questions?

Acknowledgements

Some slides were derived from Avi Silberschatz slides.