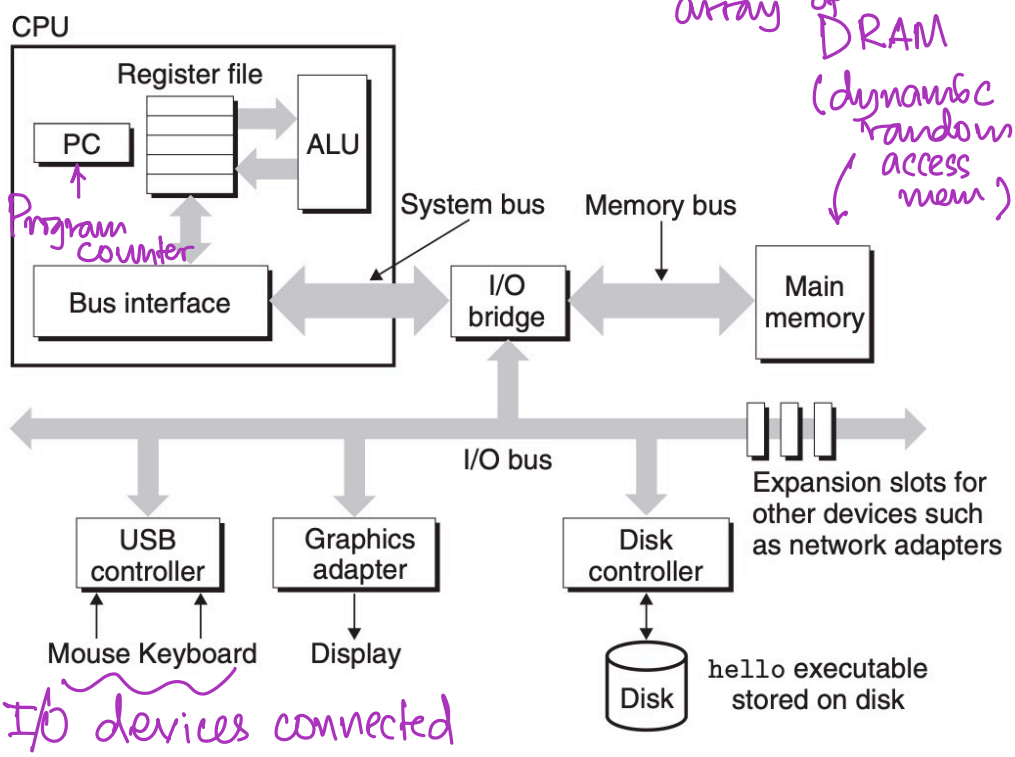
Computer Systems Organization - Spring 2023

Figure 1.4
Hardware organization of a typical system. CPU: Central Processing Unit, ALU: Arithmetic/Logic Unit, PC: Program counter, USB: Universal Serial Bus.

array of DRAM (dynamic random access mem)



Buses: Transfer fixed size chunks of bytes (called words).

32-bit/64-bit

I/O devices connected to adapters/controllers

External (to main chip/motherboard)

could be on the main chip/motherboard

DRAM ← memory is organized into linear array of bytes, each with its unique address.

Program counter (PC): Points to the current instruction in the main memory.

(From the time power is applied to the system)

Processor: Repeatedly executes the instruction pointed at by the program counter and updates the program counter to the next instruction.

Register file: Small storage device that contains a collection of word-sized registers. (with unique name).

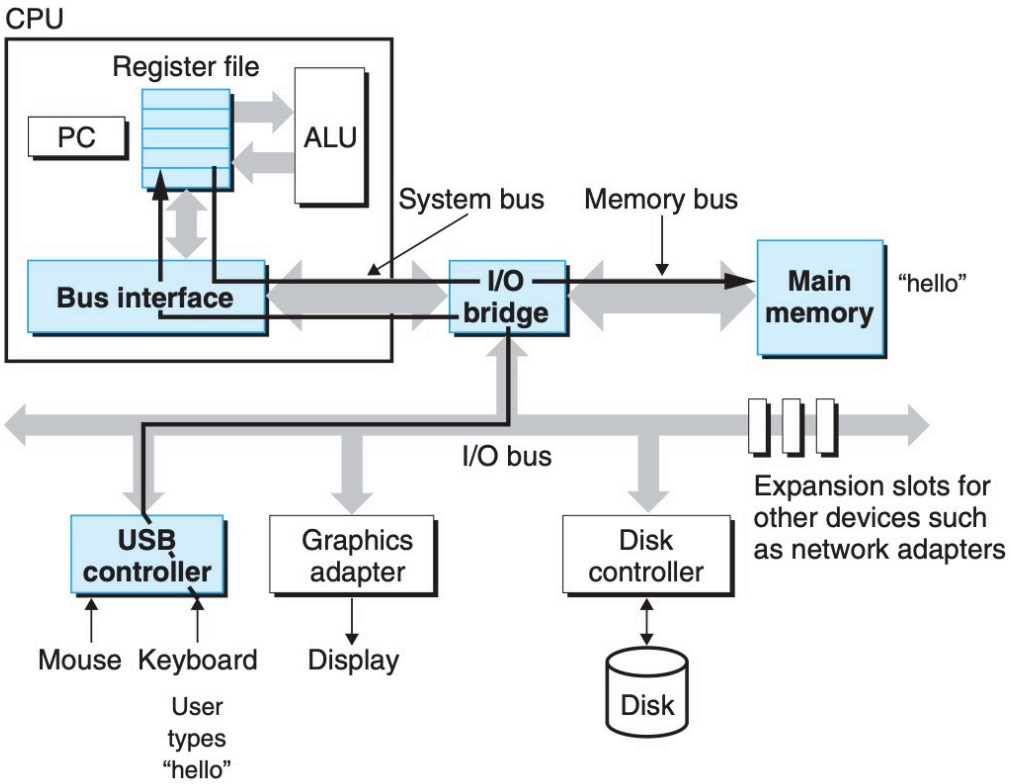
Arithmetic/Logic unit: Computes new data/address values.

CPU Simple operations

- **Load** — Copy from main memory into register
- **Store** — Copy from register to main memory
- **Operate** — Copy contents of a register to ALU, compute and Load the value into a register
- **Jump** — Extract a word from instruction, Copy the word into PC and overwrite the PC.

Figure 1.5
Reading the `hello` command from the keyboard.

→ ./hello RET



Direct Memory Access.

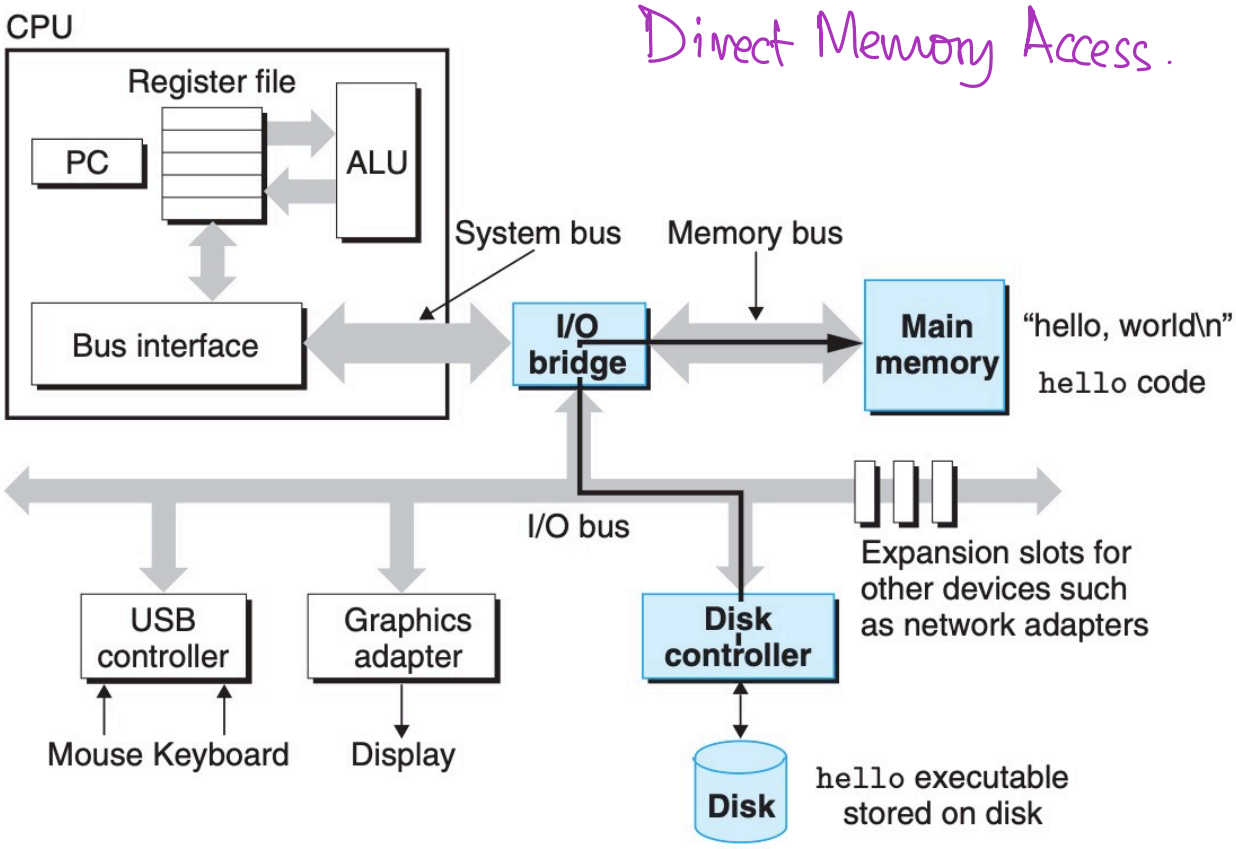
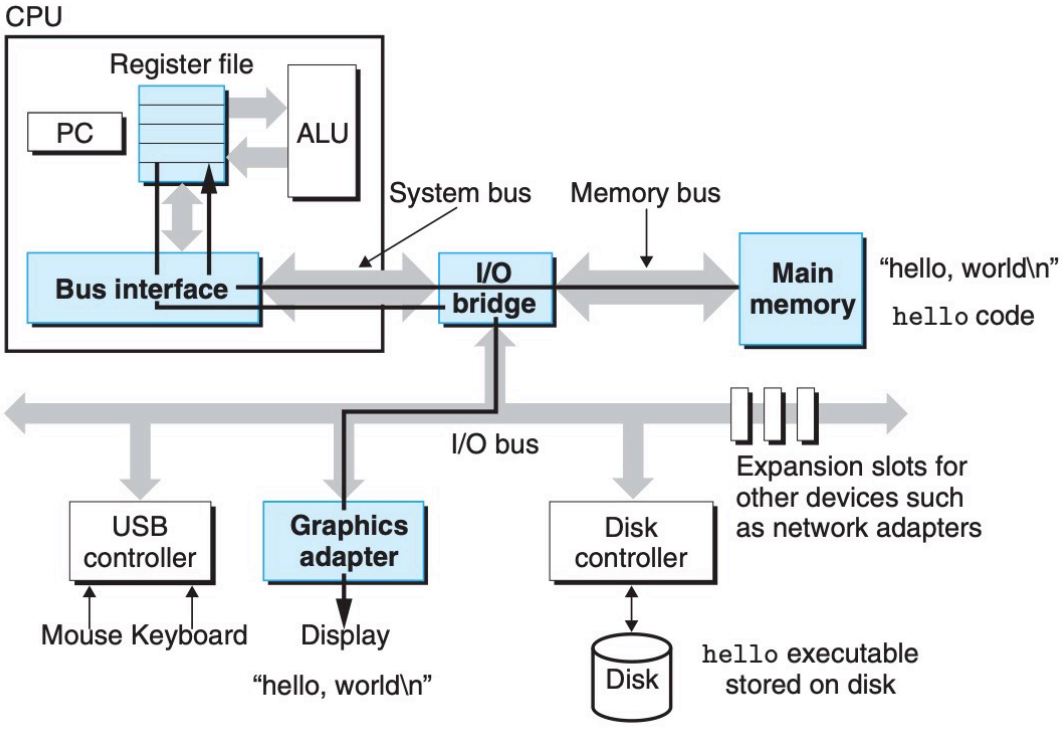


Figure 1.6 **Loading the executable from disk into main memory.**

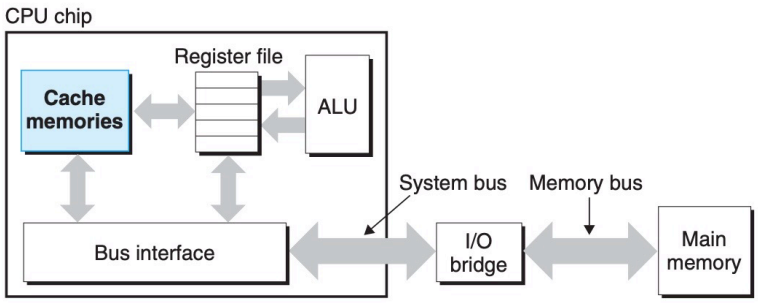
Figure 1.7
Writing the output string from memory to the display.



Bottlenecks: Processor-Memory gap.

Fix: Cache memories (temporary storage)

Implemented using SRAMs.
(inside the CPU)



DRAMs	vs SRAMs
• Need refreshing.	• Stable
• Easy to implement	• Not so simple
• Used in Main memory	• Used for caches

Hierarchy: L1 cache: stores $\lesssim 10^5$ bytes and can be accessed as fast as accessing a register file.

1 MB to 16 MB recently. L2 cache: stores $\lesssim 10^7$ bytes and may take "more" time than accessing a register file.

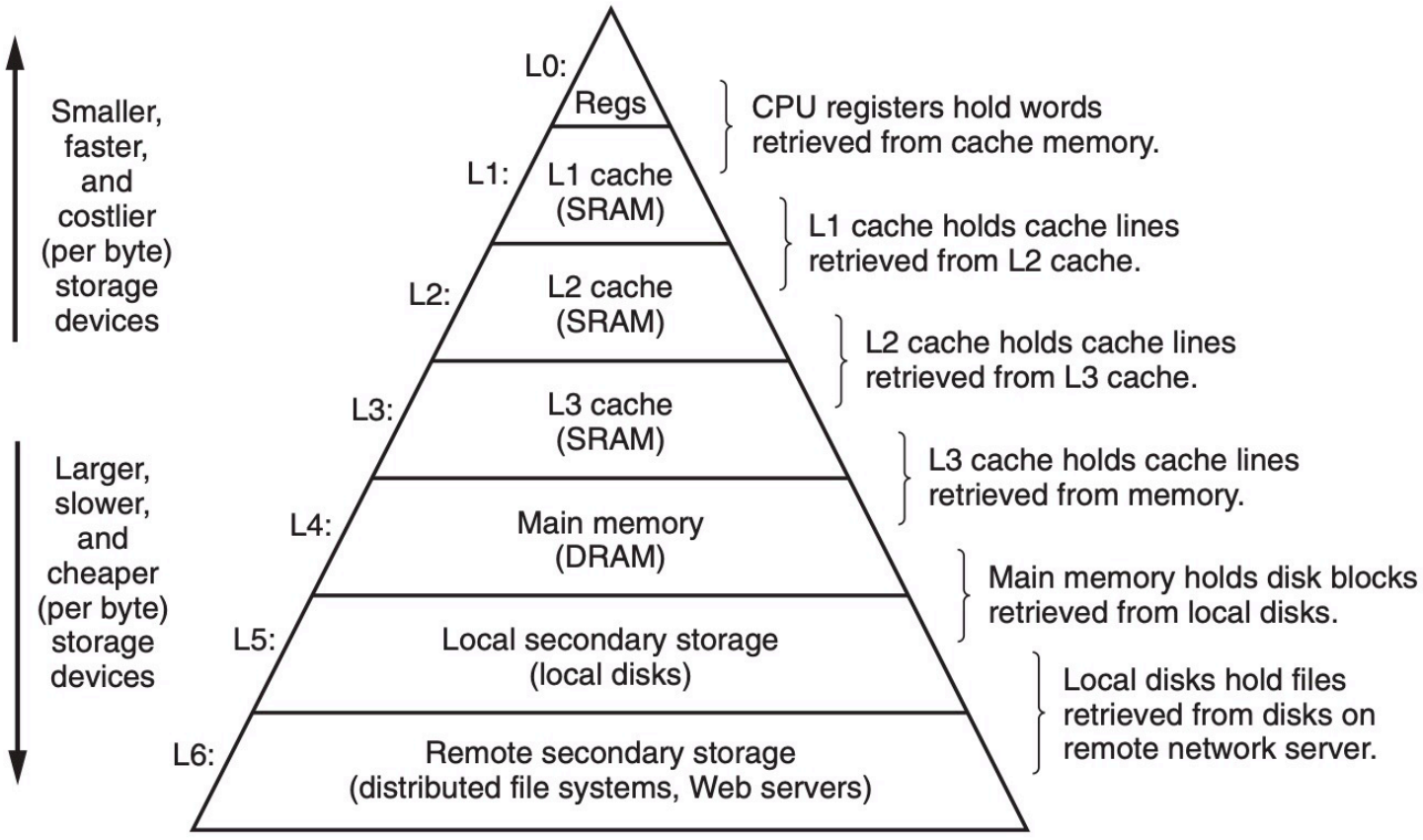
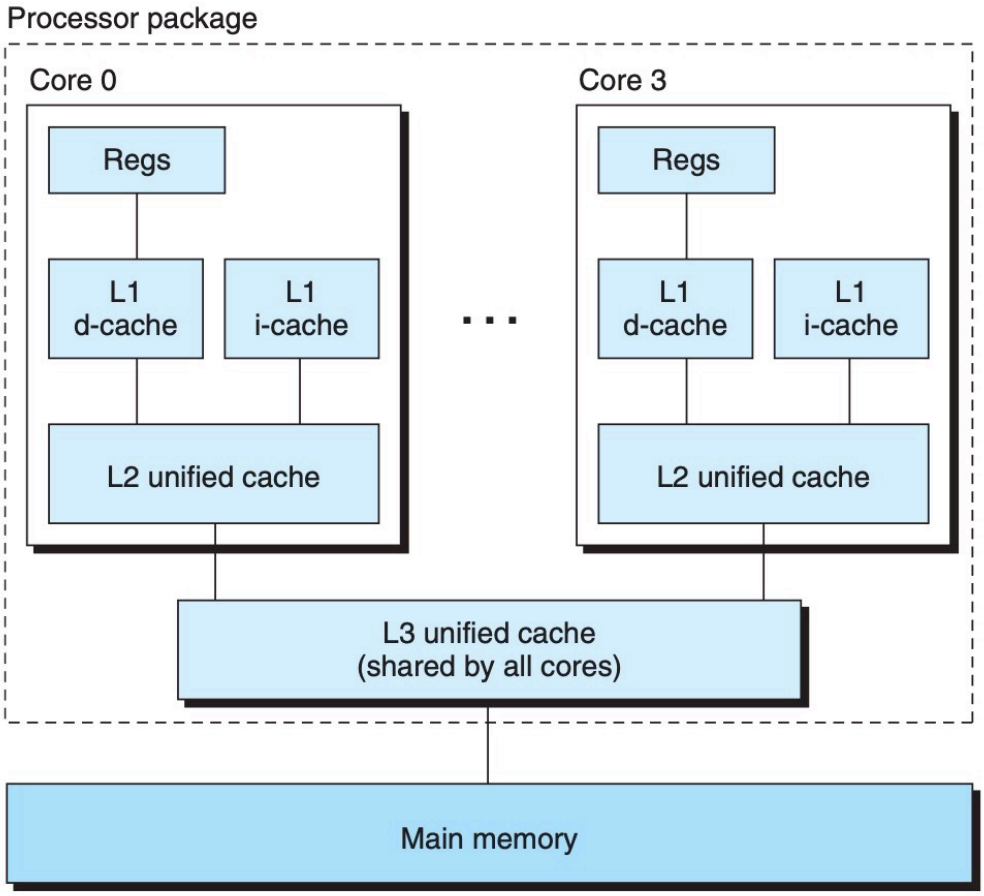


Figure 1.9 **An example of a memory hierarchy.**

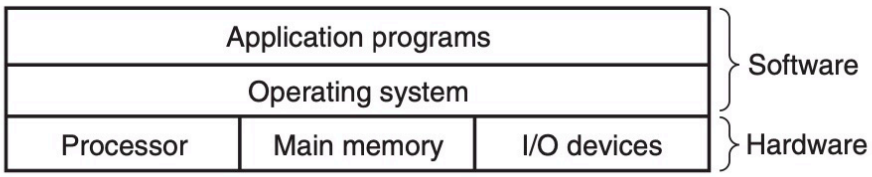
Understanding this could lead to "optimal" programs.

Figure 1.17
Intel Core i7 organization. Four processor cores are integrated onto a single chip.



Operating System

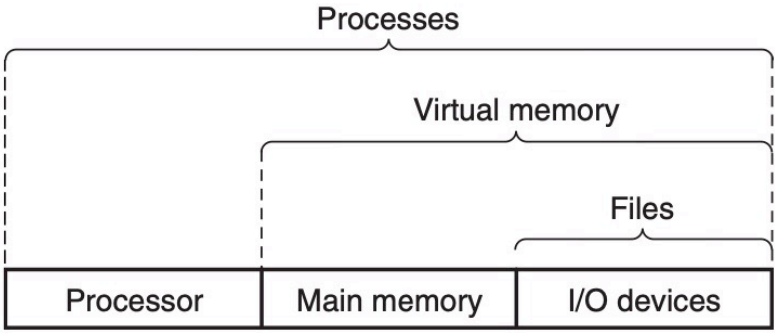
Figure 1.10
Layered view of a computer system.



Operating systems have two primary purposes:

1. to protect the hardware from misuse by runaway applications
2. to provide applications with simple and uniform mechanisms for manipulating complicated and widely different hardware devices.
   ↳ done through abstractions

Figure 1.11
Abstractions provided by an operating system.

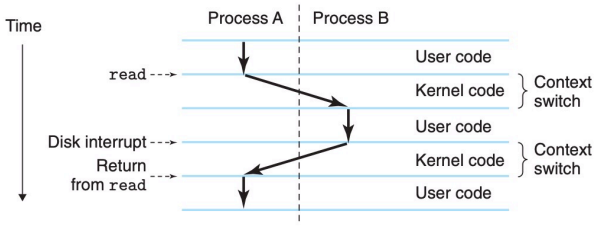


Process: Abstraction for a running program

↳ Related notion: Concurrency

↑ Could be done via context switching.

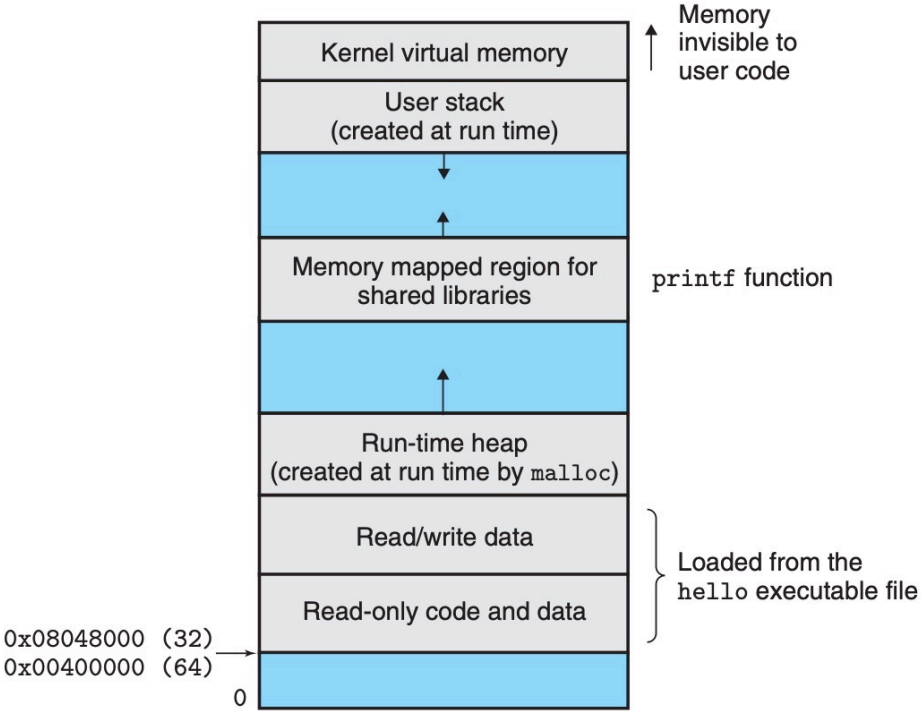
Figure 1.12
Process context switching.



Threads: Processes with a single control flow.

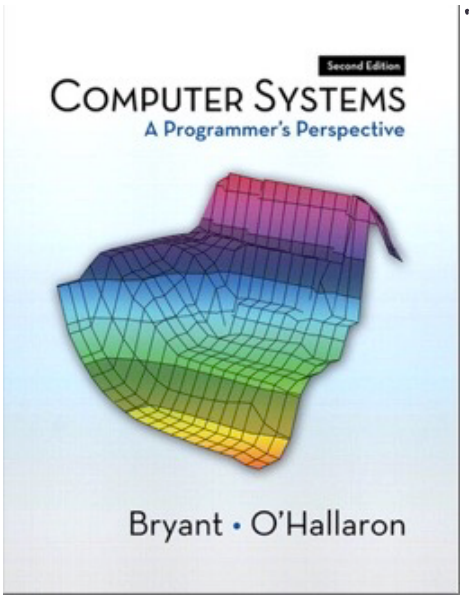
Virtual memory: Each process has a uniform view of memory known as virtual address space

Figure 1.13
Process virtual address space.

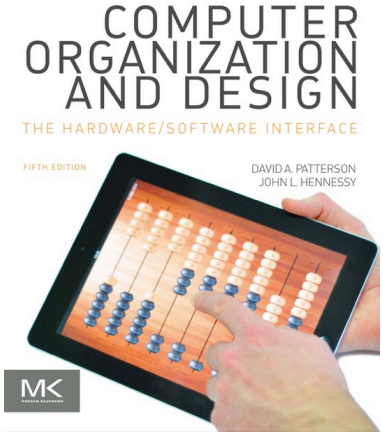


Files: Sequence of bytes.

(Typically, I/O devices are modeled as files).



Topics for the rest of the course:

- Computer arithmetic (2.1 – 2.3)
- Assembly language programming (3.4 – 3.7)
- Processor architecture and design (4.1 – 4.5)
- Memory hierarchy (6.1 – 6.4)
- System calls: Intro to process control (8.2 – 8.4).

Extra reference text: