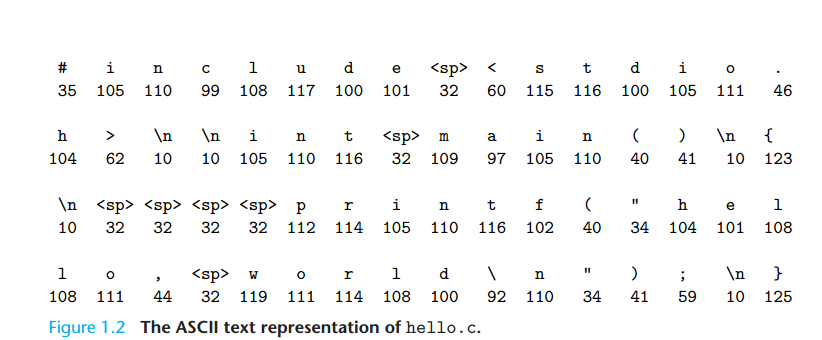
1.1 Information Is Bits + Context .

Các chương trình nguồn là một chuỗi các bit, mỗi bit có giá trị 0 hoặc 1, được tổ chức ở dạng khối 8 bit được gọi là byte. Mỗi byte đại diện cho một số ký tự văn bản trong chương trình.

các hệ thống hiện đại đại diện cho các ký tự văn bản bằng cách sử dụng tiêu chuẩn ASCII đại diện cho mỗi ký tự với một giá trị số nguyên có kích thước byte duy nhất. Exmaple.

.

Chương trình hello.c được lưu trữ trong một tệp dưới dạng một chuỗi các byte.Mỗi byte có một giá trị số nguyên tương ứng với một số ký tự.ví dụ.

byte đầu tiên có giá trị nguyên 35, tương ứng với ký tự ‘#’. Byte thứ hai có giá trị số nguyên 105, tương ứng với ký tự ‘i’. mỗi dòng văn bản được kết thúc bằng ký tự dòng mới vô hình ‘\ n’, là được đại diện bởi giá trị số nguyên 10

Origins of the C programming language

C được phát triển từ năm 1969 đến năm 1973 bởi Dennis Ritchie của Phòng thí nghiệm Bell. C gắn chặt với hệ điều hành Unix. C được phát triển ngay từ đầu với tư cách là ngôn ngữ lập trình hệ thống cho Unix. Hầu hết nhân Unix và tất cả các công cụ hỗ trợ của nó và thư viện, được viết bằng C.

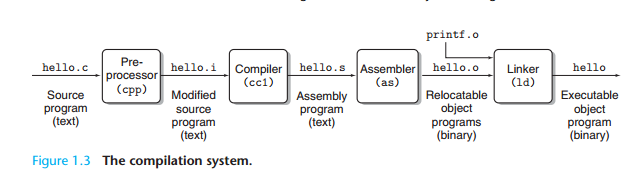
**1.2 Những chương trình được địch từ các chương trình khác qua các hình thứ khác nhau**

Để chạy hello.c trên hệ thống, các câu lệnh C riêng lẻ phải được dịch bởi các chương trình khác thành một chuỗi các lệnh ngôn ngữ máy cấp thấp,sau đó được đóng gói dưới dạng chương trình đối tượng thực thi và được lưu trữ dưới dạng nhị phân tệp đĩa

Trên hệ thống Unix, quá trình dịch từ tệp nguồn sang tệp đối tượng được thực hiện

bởi một trình điều khiển trình biên dịch

unix> *gcc -o hello hello.c*

Ở đây, trình điều khiển trình biên dịch gcc đọc tệp nguồn hello.c và dịch nó thành một tệp đối tượng thực thi hello. Quá trình dịch được thực hiện theo trình tự như hình 1.3. Các chương trình thực hiện bốn giai đoạn: preprocessor, compiler, assembler, and linker được gọi chung là hệ thống biên dịch

+ preprocessor: Bộ tiền xử lý (cpp) sửa đổi chương trình C gốc theo các lệnh bắt đầu bằng ký tự #.Ví dụ,

#include <stdio.h> trong dòng 1 của hello.c để đọc nội dung của tệp và chèn trực tiế vào văn bản chương trình

+Compilation phase: Trình biên dịch (cc1) dịch tệp văn bản hello.i thành tệp văn bản hello.s, chứa một chương trình hợp ngữ. Mỗi câu lệnh mô tả chính xác một cấp thấp hướng dẫn ngôn ngữ máy ở dạng văn bản chuẩn. Hợp ngữ cung cấp một ngôn ngữ đầu ra chung cho các trình biên dịch khác nhau cho các ngôn ngữ cấp cao khác nhau

+Assembly phase: trình hợp dịch (as) dịch hello.s thành các hướng dẫn ngôn ngữ máy, đóng gói chúng ở dạng được gọi là đối tượng có thể di dời và lưu trữ kết quả trong tệp đối tượng hello.o. Tệp hello.o là

một tệp nhị phân có các byte mã hóa hướng dẫn ngôn ngữ máy.

+Linking phase: hàm printf là một phần của thư viện C tiêu chuẩn được cung cấp bởi mọi trình biên dịch C. Printf hàm nằm trong một tệp đối tượng được biên dịch trước riêng biệt được gọi là printf.o, được hợp nhất bằng trình liên kết (ld)

**1.3:** It Pays to Understand How Compilation Systems Work

Đối với các chương trình đơn giản như hello.c, chúng ta có thể dựa vào hệ thống biên dịch để

sản xuất mã máy chính xác và hiệu quả. Tuy nhiên, có một số quan trọng lý do tại sao các lập trình viên cần hiểu cách thức hoạt động của hệ thống biên dịch:

* Tối ưu hóa hiệu suất chương trình: để đưa ra quyết định mã hóa tốt trong các chương trình C cần hiểu cơ bản về mã cấp máy và cách trình biên dịch dịch các câu lệnh C khác nhau thành mã máy Ví dụ, là một câu lệnh switch luôn hiệu quả hơn một chuỗi các câu lệnh if-else
* Hiểu lỗi thời gian liên kết: một số lỗi lập trình phức tạp nhất có liên quan đến hoạt động của trình liên kết, đặc biệt là khi bạn đang cố gắng xây dựng các hệ thống phần mềm lớn. Cái gì là

sự khác biệt giữa một biến tĩnh và một biến toàn cục?

* Tránh các lỗ hổng bảo mật. Trong nhiều năm, lỗ hổng tràn bộ đệm đã chiếm phần lớn các lỗ hổng bảo mật trong mạng và máy chủ Internet. quá ít lập trình viên hiểu được nhu cầu để hạn chế cẩn thận số lượng và hình thức dữ liệu mà họ chấp nhận từ những nơi không đáng tin cậy các nguồn

**1.4: Bộ xử lý Đọc và Giải thích Hướng dẫn Được lưu trữ trong bộ nhớ**

chương trình nguồn hello.c đã được biên dịch hệ thống thành một tệp đối tượng thực thi được gọi là hello được lưu trữ trên đĩa, được thực thi trên hệ thống Unix, chúng tôi nhập tên của nó vào một chương trình ứng dụng được gọi là shell

unix> *./hello*hello, world  
unix>

The shell là một trình thông dịch dòng lệnh in ra một lời nhắc đợi bạn nhập một dòng lệnh, và sau đó thực hiện lệnh Nếu từ đầu tiên của lệnh dòng không tương ứng với lệnh shell tích hợp sẵn, sau đó shell giả định rằng nó là tên của một tệp thực thi mà nó sẽ tải và chạy.

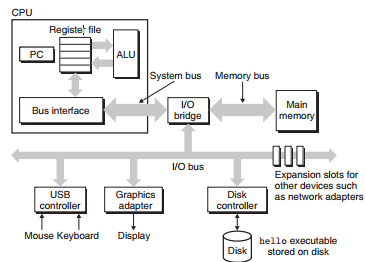
**1.4.1 Hardware Organization of a System: cấu tạo phần cứng của một hệ thống**

CPU:Bộ xử lý trung tâm

ALU: Arithmetic/Logic Unit:  là một mạch điện tử thực hiện phép tính số học và lo

PC: Program counter gic :bộ đếm chương trình

USB: Universal Serial Bus: Bus nối tiếp đa năng.



Buses:

Là một tập hợp các ống dẫn điện mang các byte thông tin qua lại giữa các thành phần, được thiết kế để chuyển các phần byte có kích thước cố định được gọi là từ

Các số byte trong một từ (kích thước từ) là một tham số hệ thống cơ bản

khác nhau giữa các hệ thống Hầu hết các máy ngày nay đều có kích thước từ là 4 byte (32 bit) hoặc 8 byte (64 bit)

I/O Devices:

Thiết bị đầu vào / đầu ra kết nối của hệ thống với thế giới bên ngoài

Example: bàn phím và chuột để người dùng nhập liệu, hiển thị và một ổ đĩa để lưu trữ lâu dài dữ liệu và chương trình

Mỗi thiết bị I/O được kết nối với bus I / O bằng bộ điều khiển hoặc bộ chuyển đổi Bộ điều khiển là chip

đặt trong chính thiết bị hoặc trên bảng mạch in chính của hệ thống (thường được gọi là bo mạch chủ). Bộ điều hợp là một thẻ cắm vào một khe cắm trên bo mạch chủ.

Main Memory

là thiết bị lưu trữ tạm thời chứa cả chương trình và dữ liệu. bộ nhớ chính bao gồm một tập hợp các bộ nhớ truy cập ngẫu nhiên động (DRAM). Về mặt logic, bộ nhớ được tổ chức như một mảng tuyến tính gồm các byte, mỗi byte có địa chỉ duy nhất (chỉ số mảng) bắt đầu từ 0.

Ví dụ: trên máy IA32 chạy Linux, dữ liệu kiểu short yêu cầu hai byte, kiểu int, float, và bốn byte dài, và nhập đôi tám byte.

Processor

Bộ xử lý trung tâm (CPU), hay đơn giản là bộ xử lý, là bộ máy diễn giải các lệnh được lưu trữ trong bộ nhớ chính. ví dụ về các hoạt động đơn giản mà CPU có thể thực hiện tại yêu cầu:

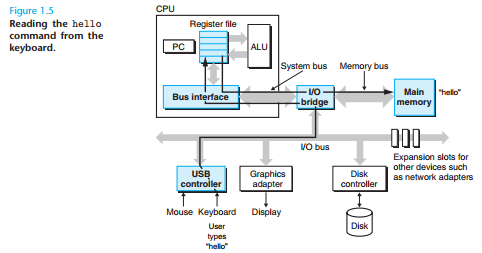
* . ***Load****:* Sao chép một byte hoặc một từ từ bộ nhớ chính vào một thanh ghi, ghi đè nội dung trước đó của sổ đăng ký.
* . Store: sao chép một byte hoặc một từ từ một thanh ghi vào một vị trí trong bộ nhớ chính,ghi đè các nội dung trước đó của vị trí đó.
* *Operate:* *Sao chép nội dung của hai thanh ghi vào ALU, thực hiện một số học hoạt động trên hai từ và lưu trữ kết quả trong một sổ đăng ký, ghi đè lên*

*nội dung trước đó của sổ đăng ký đó.*

* Jump: *Trích xuất một từ từ chính hướng dẫn và sao chép từ đó vào bộ đếm chương trình (PC), ghi đè giá trị trước đó của PC.*

**1.4.2 Running the** hello **Program**

Khi chúng ta nhập các ký tự “./hello” trên bàn phím, shell chương trình đọc từng cái vào một thanh ghi, và sau đó lưu trữ nó trong bộ nhớ,

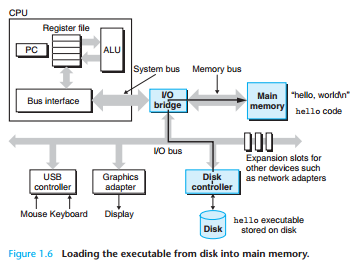


Khi chúng ta nhấn phím enter shell sẽ biết rằng chúng ta gõ lệnh xong. shell sẽ tải tệp hello thực thi bằng cách

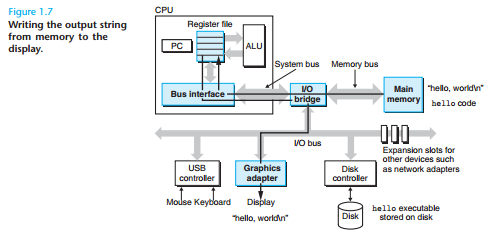
thực hiện một chuỗi hướng dẫn sao chép mã và dữ liệu trong hello tệp đối tượng từ đĩa vào bộ nhớ chính.

Dữ liệu bao gồm chuỗi ký tự “Xin chào, thế giới \ n” cuối cùng sẽ được in ra.

Sử dụng một kỹ thuật được gọi là truy cập bộ nhớ trực tiếp (DMA**)** dữ liệu truyền trực tiếp từ đĩa đến bộ nhớ chính mà không cần đi qua bộ xử lý



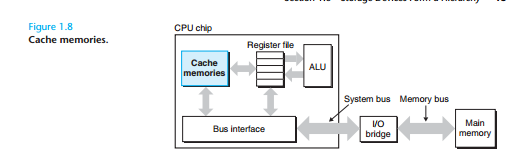
Sau khi mã và dữ liệu trong tệp đối tượng hello được tải vào bộ nhớ bộ xử lý bắt đầu thực hiện các lệnh ngôn ngữ máy sao chép các byte trong "xin chào, thế giới \ n" chuỗi từ bộ nhớ đến tệp đăng ký và từ đó đến thiết bị hiển thị,



**1.5 Caches Matter:bộ nhớ đệm Matter**

**Không phải là bộ nhớ chính của máy tính nó khá quan trọng đối vơi hiệu suất máy tính.Không giống bọ nhớ chính lớn và chậm nó là một vùng lưu trữ tạm thời nhỏ và nhanh nằm gần hoặc trên cpu để truy xuất dữ liệu**

Để giải quyết khoảng cách bộ xử lý-bộ nhớ, các nhà thiết kế hệ thống bao gồm thiết bị lưu trữ nhanh hơn được gọi là bộ nhớ cache đóng vai trò như

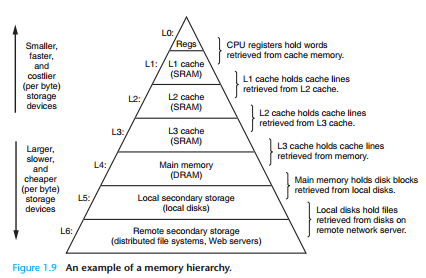
các khu vực tạm thời cho thông tin mà bộ xử lý có thể cần trong tương lai gần.

Nộ nhớ đệm sử dụng SRAM tốc độ cao

có ba mức bộ nhớ cache:L1 là bộ nhớ đệm chính cực nhanh và nhỏ thường được nhúng vào CPU dưới dạn bộ nhớ đệm CPU

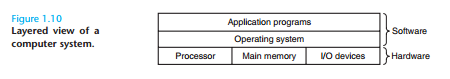
,L2 là bộ đếm thứ cấp thường chứa nhiều hơn bộ đếm L1có thể được nhúng như L1 hoặc nằm trên một chip riêng biệt với bus hệ thống thay thế tốc độ cao kết nối bộ nhớ đệm và CPU

L3.là bộ nhớ chuyên dụng được phát triển để cải thiện hiệu suất của L1,L2.nó chậm hơn đáng kể so với L1 và L2 nhưng nó nhanh gấp đôi DRAM



**1.7 The Operating System Manages the Hardware**

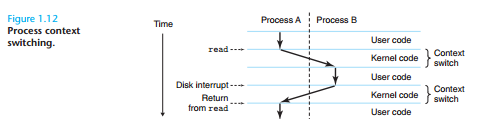
Quay lại ví dụ xin chào của chúng tôi. Khi shell được tải và chạy chương trình hello,và khi chương trình xin chào in ra thông báo của nó, cả hai chương trình đều không truy cập vào trực tiếp bàn phím, màn hình, đĩa hoặc bộ nhớ chính. Thay vào đó, họ dựa vào dịch vụ do hệ điều hành cung cấp. Chúng ta có thể coi hệ điều hành là một lớp phần mềm xen kẽ giữa chương trình ứng dụng và phần cứng,



**1.7.1 Processes:quy trình**

Khi một chương trình như hello chạy trên một hệ thống hiện đại, hệ điều hành cung cấp nghĩ rằng chương trình là chương trình duy nhất chạy trên hệ thống Các chương trình dường như có quyền sử dụng độc quyền cả bộ xử lý, bộ nhớ chính và I / O thiết bị

Hệ điều hành theo dõi tất cả các thông tin trạng thái mà quá trình cần để chạy Trạng thái này, được gọi là ngữ cảnh, bao gồm thông tin như giá trị hiện tại của PC, tệp đăng ký và nội dung của bộ nhớ chính.



1.7.2 Threads

Mặc dù chúng ta thường nghĩ về một quy trình là có một luồng điều khiển duy nhất, nhưng trong hiện đại hệ thống một quy trình thực sự có thể bao gồm nhiều đơn vị thực thi, được gọi là luồng, Luồng là một mô hình lập trình ngày càng quan trọng vì

yêu cầu đồng thời trong các máy chủ mạng, vì dễ dàng chia sẻ dữ liệu hơn

giữa nhiều luồng hơn là giữa nhiều quá trình và bởi vì các luồng thường hiệu quả hơn các quy trình.

**1.7.3 Virtual Memory** :bộ nhớ ảo

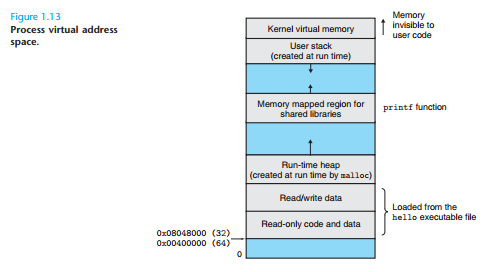
cung cấp cho mỗi quá trình ảo có quyền sử dụng độc quyền của bộ nhớ chính. Mỗi quy trình có cùng một quan điểm thống nhất về bộ nhớ, được gọi là không gian địa chỉ ảo của nó. Các quy trình của Linux được thể hiện trong Hình 1.13

Trong Linux, vùng trên cùng của không gian địa chỉ được dành riêng cho mã và dữ liệu

trong hệ điều hành chung cho tất cả các quy trình. Vùng thấp hơn của

không gian địa chỉ lưu giữ mã và dữ liệu được xác định bởi quy trình của người dùng. Lưu ý rằng địa chỉ trong hình tăng dần từ dưới lên trên.

Không gian địa chỉ ảo được nhìn thấy bởi mỗi quá trình bao gồm một số vùng được xác định cụ thể, mỗi vùng có một mục đích cụ thể

  
-Heap: Các vùng mã và dữ liệu được theo sau ngay lập tức bởi heap thời gian chạy. Không giống như các vùng mã và dữ liệu, có kích thước cố định khi quá trình bắt đầu đang chạy, heap mở rộng tại thời gian chạy

*-Shared libraries***:** Gần giữa không gian địa chỉ là một khu vực chứa mã và dữ liệu cho các thư viện được chia sẻ như thư viện tiêu chuẩn C và toán học thư viện. Là một công cụ mạnh mẽ nhưng hơi khó

-Stack: Ở đầu không gian địa chỉ ảo của người dùng, dùng trình biên dịch sử dụng để thực hiện các lời gọi hàm

-Kernel virtual memory: là một phần của hệ điều hành, vùng trên cùng của không gian địa chỉ Các chương trình ứng dụng không được phép đọc hoặc ghi nội dung của vùng này hoặc để gọi trực tiếp các hàm được định nghĩa.

**1.7.4 Files**

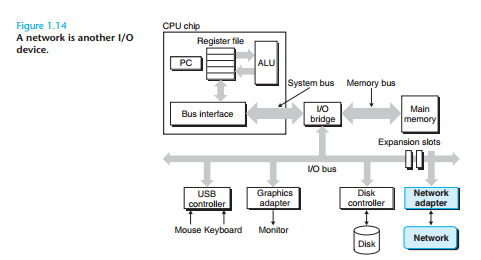
**L**à một chuỗi các bytes, Mọi thiết bị I / O,

bao gồm đĩa, bàn phím, màn hình và thậm chí cả mạng, được mô hình hóa dưới dạng tệp. Tất cả đầu vào và đầu ra trong hệ thống được thực hiện bằng cách đọc và ghi tệp, sử dụng một tập hợp nhỏ các lệnh gọi hệ thống được gọi là Unix I / O.

Nó cung cấp cho các ứng dụng một cái nhìn thống nhất về tất cả các thiết bị I / O đa dạng có thể được chứa trong hệ thống.

**1.8 Systems Communicate with Other Systems  
Using Networks**

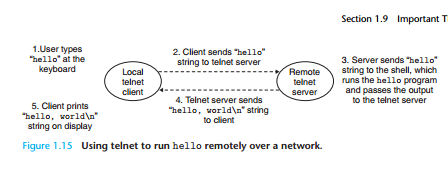
**Trong thực tế, các hệ thống hiện đại thường được liên kết đến các hệ thống khác theo mạng.** **Từ quan điểm của một hệ thống riêng lẻ,mạng có thể được xem như một thiết bị I / O khác**

****

**Với sự ra đời của các mạng toàn cầu như Internet, việc sao chép thông tin từ máy này sang máy khác đã trở thành một trong những ứng dụng quan trọng nhất của hệ thống máy tính.**

**ví dụ: các ứng dụng như email, nhắn tin tức thì, World Wide Web, FTP và telnet đều dựa trên khả năng sao chép thông tin qua mạng**

**Quay trở lại ví dụ hello giả sử ,sử dụng một ứng dụng khách telnet chạy trên máy cục bộ để kết nối máy chủ telnet từ xa.sau khi đăng nhập và chạy một trình báo từ xa đang chờ một lệnh đầu vào ,việc chạy chương trình hello từ xa gồm 5 bước cơ bản:**

****

**Sau khi nhập chuỗi vào ứng dụng telnetmays khách sẽ gửi chuỗi đến máy chủ telnet sau đó chuyển nó cùng với chương trình shell từ xa**

**remote shell chạy chương trình hello và chuyển dòng đầu ra trở lại telnet máy chủ Cuối cùng, máy chủ telnet chuyển tiếp chuỗi đầu ra qua mạng tới ứng dụng khách telnet, in chuỗi đầu ra trên thiết bị đầu cuối cục bộ**

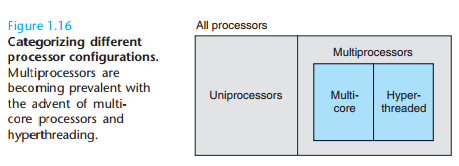
**1.9 Important Themes**

**1.9.1 Concurrency and Parallelism**

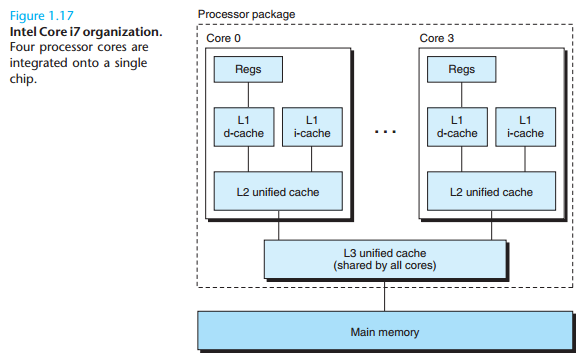
**ba cấp độ trong hệ thống phân cấp:**

**+** Thread-Level Concurrency:

**Hình 1.16 cho thấy sự phân loại của những các loại bộ xử lý**

****

**Bộ xử lý đa lõi có một số CPU được tích hợp trên một chip mạch tích hợp duy nhất.** **Bộ xử lý Intel Core i7, trong đó chip vi xử lý có bốn lõi CPU, mỗi lõi với bộ nhớ đệm L1 và L2 của riêng nó nhưng chia sẻ mức bộ nhớ đệm cao hơn cũng như giao diện với bộ nhớ chính. Các chuyên gia trong ngành dự đoán rằng họ sẽ có thể có hàng chục và cuối cùng là hàng trăm lõi trên một con chip.**

****

**bộ xử lý yêu cầu khoảng 20.000 chu kỳ xung nhịp để thay đổi giữa các luồng khác nhau,** **một bộ xử lý siêu phân luồng quyết định luồng nào của nó sẽ thực thi trên cơ sở chu kỳ theo chu kỳ. cho phép CPU tận dụng tốt hơn quá trình xử lý của nó tài nguyên**

**Ví dụ: bộ xử lý Intel Core i7 có thể có mỗi lõi thực hiện hai luồng và**

**vì vậy một hệ thống bốn lõi thực sự có thể thực thi tám luồng song song.**

**Việc sử dụng đa xử lý cải thiệ hiệu suất theo 2 cách:giảm nhu cầu mo phỏng và thực hiện nhiều tác vụ**

Instruction-Level Parallelism:

**Ở mức độ trừu tượng thấp hơn các bộ xử lý hiện đại có thể thực thi nhiều các lệnh tại một thời điểm**

**Bộ xử lý có thể duy trì tốc độ thực thi nhanh hơn một lệnh cho mỗi chu trình được gọi là bộ xử lý siêu cực,** **hầu hết các bộ vi xử lý hiện đại đều hỗ trợ hoạt động superscalar.**

Single-Instruction, Multiple-Data (SIMD) Parallelism

# Một lệnh đơn, nhiều dữ liệu:

Ở cấp thấp nhất, nhiều bộ vi xử lý hiện đại có phần cứng đặc biệt cho phép

một lệnh duy nhất để thực hiện song song nhiều thao tác, một chế độ được gọi là chế độ song song một lệnh, nhiều dữ liệu hoặc “SIMD”

ví dụ, các thế hệ bộ xử lý Intel và AMD gần đây có hướng dẫn

có thể thêm bốn cặp số dấu phẩy động chính xác đơn (float kiểu dữ liệu C)

song song.

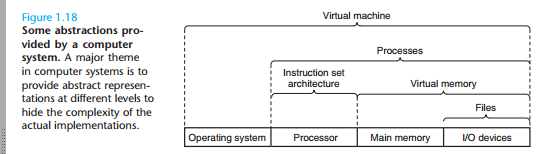
Các hướng dẫn SIMD này hầu hết được cung cấp để tăng tốc các ứng dụng

xử lý dữ liệu hình ảnh, âm thanh và video.

Mặc dù một số trình biên dịch cố gắng tự động trích xuất song song SIMD từ các chương trình C, một phương pháp đáng tin cậy hơn là viết chương trình sử dụng kiểu dữ liệu vectơ đặc biệt được hỗ trợ trong trình biên dịch như gcc.

**1.9.2 The Importance of Abstractions in Computer Systems**

một số tính năng trừu tượng trong các hệ thống máy tính, như được chỉ ra trong Hình 1.18



**1.10 Summary**

Hệ thống máy tính bao gồm phần cứng và phần mềm hệ thống hợp tác để chạy các chương trình ứng dụng. Thông tin bên trong máy tính được biểu diễn dưới dạng các nhóm bit được diễn giải theo nhiều cách khác nhau, tùy thuộc vào ngữ cảnh. Các chương trình được các chương trình khác dịch sang các dạng khác nhau, bắt đầu là Văn bản ASCII và sau đó được dịch bởi trình biên dịch và trình liên kết thành tệp thực thi nhị phân các tập tin.

Bộ xử lý: đọc và giải thích các lệnh nhị phân được lưu trữ trong main memory

Các thiết bị I / O và thanh ghi CPU, các thiết bị lưu trữ trong một hệ thống được sắp xếp

trong một hệ thống phân cấp, với thanh ghi CPU ở trên cùng, tiếp theo là nhiều cấp bộ nhớ cache phần cứng, bộ nhớ chính DRAM và ổ lưu trữ.

các thiết bị cao hơn trong phân cấp nhanh hơn và đắt hơn trên mỗi bit những người thấp hơn trong hệ thống phân cấp

Các thiết bị lưu trữ cao hơn trong phân cấp sẽ phân phát dưới dạng bộ nhớ đệm cho các thiết bị thấp hơn trong phân cấp.

Có thể tối ưu hóa hiệu suất của các chương trình Cbằng cách khai thác bộ nhớ hê thống cấp bậc

- hệ điều hành đóng vai trò trung gian giữa ứng dụng và phần cứng.

1:các file là abstractions cho thiết vị I/O

2 Virtual memory is an abstraction for both main memory and disks.

3: Processes are abstractions for the processor, main memory, and I/O devices.

Mạng cung cấp các cách để hệ thống máy tính giao tiếp với nhau.