

KIẾN TRÚC MÁY TÍNH



TS. Đặng Hoàng Long
Khoa Công nghệ thông tin
Khoa học máy tính

Email: longdh@ptit.edu.vn

THÔNG TIN GIẢNG VIÊN



Lecturer

Faculty of Information Technology,
PTIT

About me

I am Long Hoang Dang (also known as Đặng Hoàng Long). I studied for a Bachelor of Information Technology at the Posts and Telecommunications Institute of Technology (PTIT) from 2014 to 2019, graduating with distinction as the top-ranked student, under the supervision of Prof. [Tu Minh Phuong](#). In 2020, I received a Deakin University Postgraduate Research Scholarship (DUPRS) and started my PhD at the [Applied Artificial Intelligence Institute \(A2I2\)](#) at Deakin University, Australia under the supervision of Prof. [Truyen Tran](#), Dr. [Thao Minh Le](#), Dr. [Vuong Le](#) and Prof. [Svetha Venkatesh](#). I completed my PhD in early 2024 and have been working as a Lecture at PTIT. My research interests include Generative Models, Vision and Language Reasoning, and Domain Generalization.


News

2024-2025 Intake: Currently, I am seeking undergraduate students (2nd, 3rd, and 4th year) interested in working with me on vision and language reasoning, generative model, and domain generalization. Selected students will have the opportunity to collaborate with experts in these fields and potentially submit their work to prestigious AI conferences and journals. If you are interested, please email me your CV [here](#).

If you're interested in applying, please see the detailed instructions at this link.



TÀI LIỆU THAM KHẢO

1. Hoàng Xuân Dậu, Bài giảng môn Kiến trúc Máy tính, 2010;
 2. Phạm Hoàng Duy, Bài giảng môn Kỹ thuật Vi xử lý, 2011;
 3. Stallings W., Computer Organization and Architecture, Prentice – Hall 2013.
 4. Ata Elahi, Computer Systems: Digital design, fundamentals of architecture and assembly language, 1st edition, Springer, 2018
- 

ĐIỂM THÀNH PHẦN


- Điểm chuyên cần: 10%
 - Nghỉ học 1 ca (2 tiết) – trừ 2 điểm
 - Phát biểu và trả lời đúng câu hỏi được cộng 1 điểm.
 - Nghỉ quá 20% số buổi – không đủ điều kiện dự thi cuối kì
- Điểm kiểm tra giữa kỳ: 10%
- Bài tập lớn: 20%
- Thi cuối kỳ: 60%

ĐỀ TÀI BÀI TẬP LỚN

Chủ đề loại 1: Lập trình game đơn giản


- ▶ Đề tài 1: Lập trình game Snake trên Emu8086
- ▶ Đề tài 2: Lập trình game Tic Tac Toe trên Emu8086
- ▶ Đề tài 3: Lập trình game Rapid Roll
- ▶ Đề tài 4: Lập trình game Pong
- ▶ Đề tài 5: Lập trình game Ballon Shooting
- ▶ Đề tài 6: lập trình một game bất kỳ khác các game trên

Chủ đề loại 2: Lập trình mô phỏng

- ▶ Đề tài 7: Mô phỏng hiển thị nhiều đèn led đơn nhấp nháy
 - ▶ Đề tài 8: Mô phỏng hiển thị đèn led 7 thanh để đếm số (từ 0 - 9)
 - ▶ Đề tài 9: Mô phỏng hệ thống đèn giao thông
 - ▶ Đề tài 10: Mô phỏng giao tiếp với LCD
- 

ĐỀ TÀI BÀI TẬP LỚN

Chủ đề loại 3: Lập trình Ứng dụng

- ▶ Đề tài 11: Lập trình ứng dụng máy tính cầm tay - Mini Calculator.
 - ▶ Đề tài 12: Lập trình ứng dụng tính toán chỉ số BMI - BMI Calculator.
 - ▶ Đề tài 13: Lập trình ứng dụng học bảng chữ cái cho trẻ em - Child Learning (bảng chữ cái ABC và từ minh hoạ tương ứng).
 - ▶ Đề tài 14: Lập trình ứng dụng cho một bài Quiz đơn giản - Quiz System.
 - ▶ Đề tài 15: Lập trình ứng dụng hệ thống hoá đơn nhà hàng - Restaurant Billing System
- 

YÊU CẦU BÀI TẬP LỚN

Báo cáo phần Bài tập lớn:

► Nội dung báo cáo

- Giới thiệu đề tài: Trình bày khái quát về đề tài/ bài toán giải quyết
- Nội dung chính của đề tài: Miêu tả các giải thuật/ Lời giải lựa chọn
- Miêu tả chương trình: Các hàm chính của chương trình, Miêu tả dữ liệu đầu ra, đầu vào, Công việc mà hàm thực hiện
- Miêu tả giao diện chương trình (Optional): Điểm cộng cho chương trình có giao diện đồ họa
- Tài liệu tham khảo (Optional): Nếu chương trình có tham khảo từ các nguồn thì phải chỉ rõ những phần tham khảo. Nếu không chỉ ra sẽ nhận 0 điểm

YÊU CẦU BÀI TẬP LỚN

- Mỗi sinh viên phải trình bày nội dung bài tập và trình bày các kết quả
- Thực nghiệm thu được bao gồm Mã nguồn, Giao diện trên Emu8086, Kết quả chạy code trên Screen Emu8086, Các giao diện trình bày và nhận xét.
- Báo cáo phần nghiên cứu làm việc theo nhóm sẽ báo cáo chung cả nhóm (mỗi nhóm 7 - 8 người) - Lựa chọn 2 trong 3 chủ đề để thực hiện.
- Slides trình bày trong khoảng 17-20 phút. Tất cả các thành viên phải lên thuyết trình (30% điểm)
- Thành viên trong nhóm không nắm rõ về nội dung đề tài của nhóm: Cả nhóm bị trừ 1 điểm mỗi lần

Tại sao chúng ta cần phải học môn Kiến trúc máy tính?





Chương 1: Giới thiệu chung

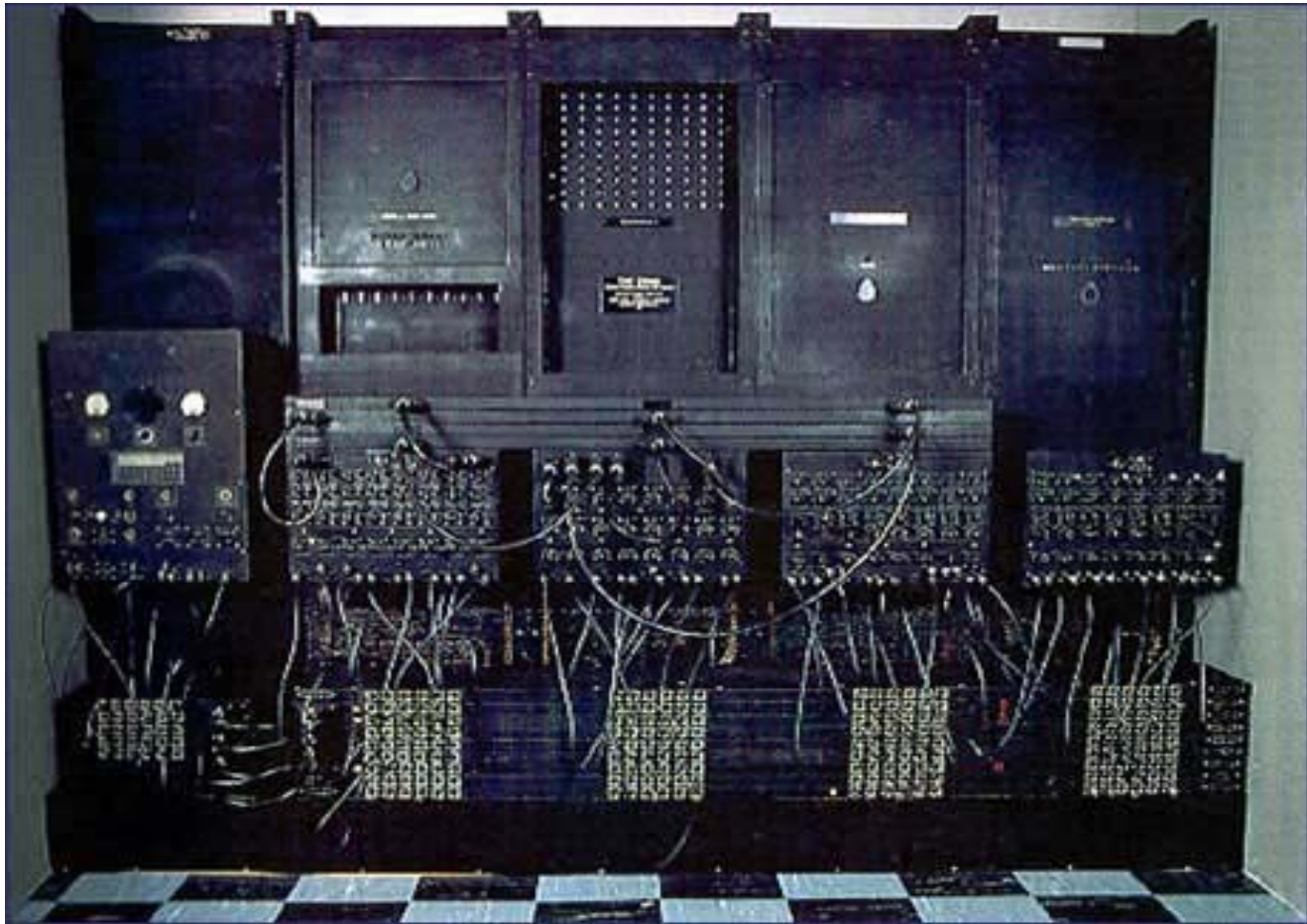
Chương 1: Nội dung chính

1. Lịch sử phát triển máy tính
2. Khái niệm kiến trúc và tổ chức máy tính
3. Cấu trúc và chức năng của máy tính
4. Kiến trúc Von Neumann
5. Kiến trúc Harvard
6. Biểu diễn dữ liệu trong máy tính

Lịch sử phát triển máy tính

- Chia thành 5 thế hệ dựa trên sự phát triển mạch điện tử
- Thế hệ 1 (1944-1959):
 - Sử dụng bóng đèn điện tử
 - Dùng băng từ làm các thiết bị đầu vào/ ra
 - Mật độ tích hợp linh kiện: 1000 linh kiện/ foot³ (1 foot= 30.48 cm)
 - Ví dụ: ENIAC - Electronic Numerical Integrator and Computer, 1946, giá 500,000 USD.

Lịch sử phát triển máy tính - ENIAC



Lịch sử phát triển máy tính

- Thế hệ thứ 2(1960-1964):
 - Sử dụng transistors
 - $\sim 100,000$ linh kiện/ foot³
 - Ví dụ: UNIVAC 1107, UNIVAC III, IBM 7070, 7080, 7090, 1400 series, 1600 series. (1951, đầu tiên giá \$159K, sau đó UNIVAC 1 giá hơn 1 triệu \$)

Lịch sử phát triển máy tính– UNIVAC



Lịch sử phát triển máy tính

- Thế hệ thứ 3 (1964-1975):
 - Sử dụng mạch tích hợp (IC)
 - ~ 10 triệu linh kiện/ foot³
 - Ví dụ: UNIVAC 9000 series, IBM System/360, System 3, System 7

Lịch sử phát triển máy tính– UNIVAC 9400



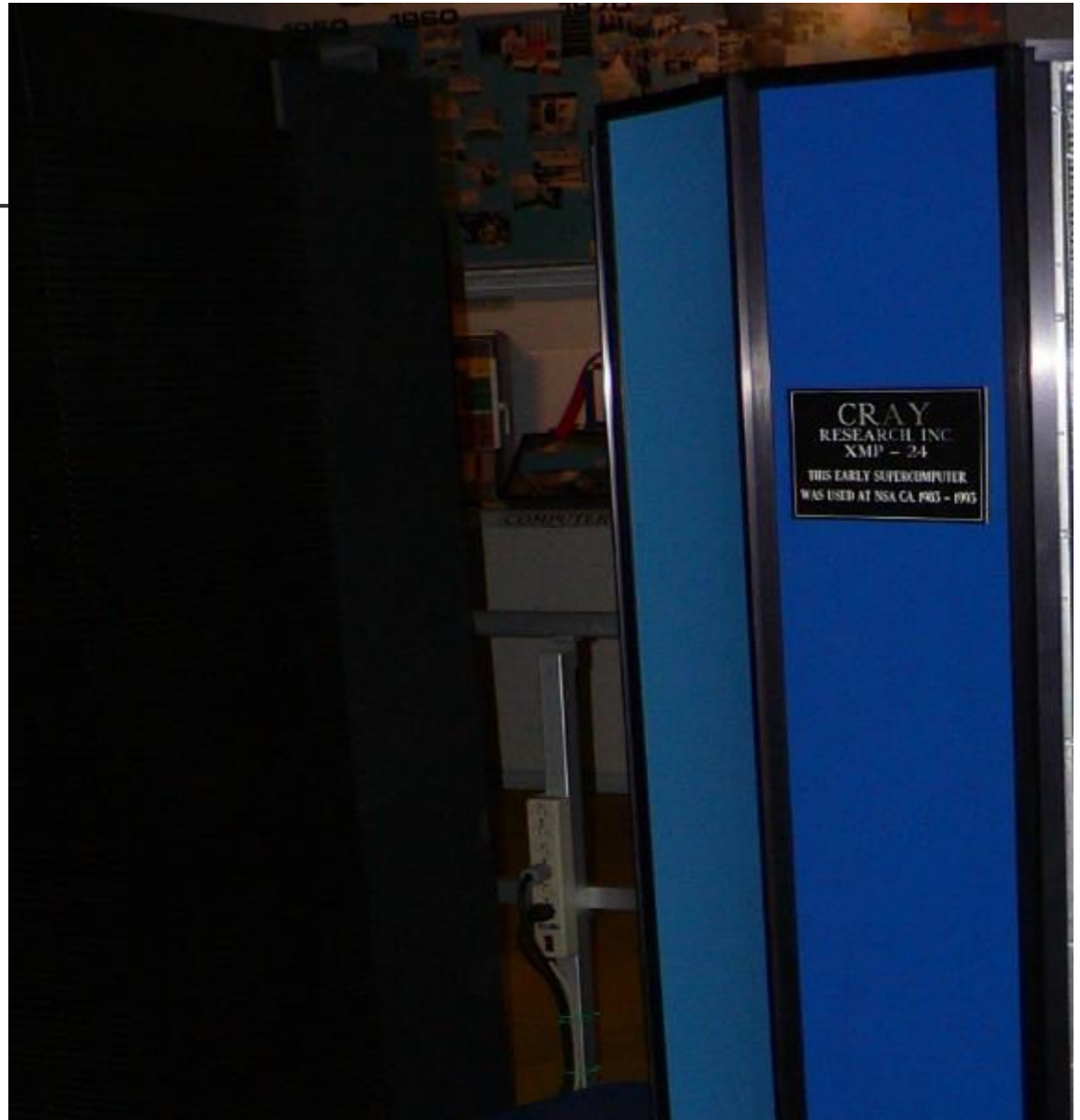
Lịch sử phát triển máy tính

□ Thế hệ 4(1975-1989):

- Sử dụng LSI – Large Scale Integrated Circuit
- ~ 1 tỷ linh kiện / foot³
- Ví dụ: IBM System 3090, IBM RISC 6000, IBM RT, Cray 2 XMP

History of computers

Cray 2 XMP



Lịch sử phát triển máy tính

- Thế hệ thứ 5 (1990- nay):
 - Sử dụng VLSI – Very Large Scale Integrated Circuit
 - $<10\text{nm} - 0.045\mu\text{m}$
 - Ví dụ: Pentium II, III, IV, M, D, Core Duo, Core 2 Duo, Core Quad,...
 - Hỗ trợ xử lý song song
 - Hiệu năng rất cao
 - Kết hợp xử lý giọng nói và hình ảnh

Kiến trúc



Kiến trúc ...



Bài học

- So sánh 2 kiến trúc trên
- Giống nhau
 - Cùng chức năng một ngôi nhà (Ăn, ngủ, sinh hoạt)
- Khác nhau
 - Vật liệu khác nhau
 - Cách bố trí
 - Giá thành
- Bài học
 - Sức sáng tạo
 - Bắt đầu từ một nhu cầu đơn giản -> cải tiến và hoàn thiện

1. Kiến trúc và tổ chức máy tính

- Kiến trúc máy tính (computer architecture): là khoa học về lựa chọn và kết nối các thành phần phần cứng của máy tính nhằm đạt yêu cầu:
 - Hiệu năng: càng nhanh càng tốt
 - Chức năng: nhiều chức năng
 - Giá thành: càng rẻ càng tốt
- Tổ chức máy tính (computer organization): là khoa học nghiên cứu các thành phần của máy tính và phương thức làm việc của chúng dựa trên kiến trúc cho trước

Ví dụ

- ❑ Computer Architecture: IBM Thinkpad, Iphone, Android Phones → Thiết kế hệ thống dòng sản phẩm
- ❑ Computer Organization → Các thế hệ máy khác nhau với cùng kiến trúc phần cứng
 - Các đời Thinkpad khác nhau: T40, T50, ..
 - Iphone 4, 4s, 5, 5s, 6, 6s, 7, 7s, 8, X, 11, 12, 13
 - Samsung S2, S3, S4, S5, S6, S7, S8, S10, Z...

1. Kiến trúc và tổ chức máy tính

□ 3 thành phần cơ bản của kiến trúc máy tính

1. Kiến trúc tập lệnh (ISA): là hình ảnh trừu tượng của máy tính ở mức ngôn ngữ máy (hoặc hợp ngữ), bao gồm:
 - Tập lệnh
 - Các chế độ địa chỉ bộ nhớ
 - Các thanh ghi
 - Khuôn dạng địa chỉ và dữ liệu

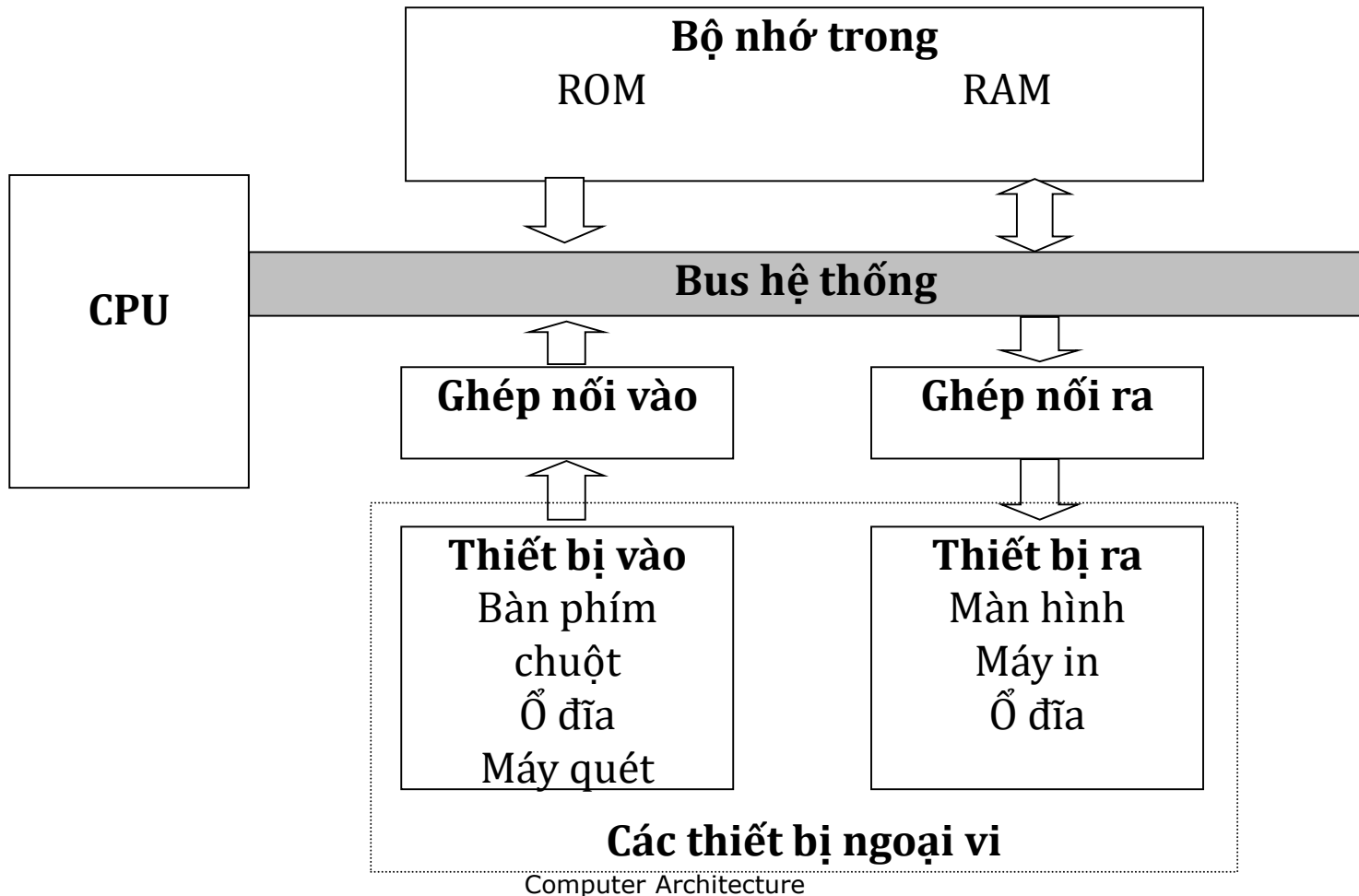
1. Kiến trúc và tổ chức máy tính

2. Vi kiến trúc (microarchitecture): còn được gọi là tổ chức máy tính, mô tả về hệ thống ở mức thấp, liên quan tới:
- ▣ Các thành phần phần cứng kết nối với nhau như thế nào
 - ▣ Các thành phần phần cứng phối hợp, tương tác với nhau như thế nào để thực hiện tập lệnh

1. Kiến trúc và tổ chức máy tính

3. Thiết kế hệ thống, bao gồm tất cả các thành phần phần cứng khác trong hệ thống máy tính, ví dụ:
 - ❑ Các hệ thống kết nối như bus và chuyển mạch
 - ❑ Mạch điều khiển bộ nhớ, cấu trúc phân cấp bộ nhớ
 - ❑ Các kỹ thuật giảm tải cho CPU như truy cập trực tiếp bộ nhớ
 - ❑ Các vấn đề như đa xử lý

2. Cấu trúc và các thành phần chức năng



2. Cấu trúc và các thành phần chức năng

□ Bộ xử lý trung tâm (CPU):

■ Chức năng:

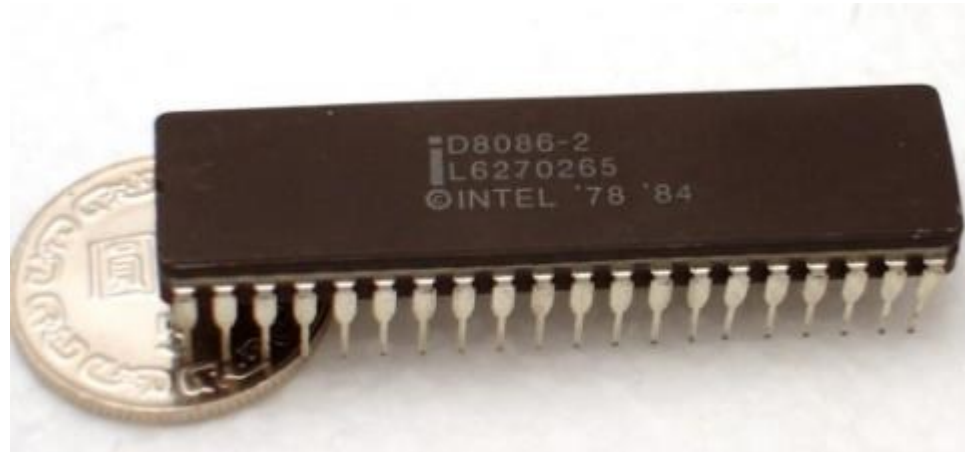
- Đọc lệnh từ bộ nhớ
- Giải mã và thực hiện lệnh

■ Bao gồm:

- Khối điều khiển (CU: Control Unit)
- Khối tính toán số học và logic (ALU: Arithmetic and Logic Unit)
- Các thanh ghi (Registers)
- Bus trong CPU

CPU

**Vi xử lý Intel
8086 (1978)**



**Vi xử lý Intel
Core 2 Duo
(2006)**



2. Cấu trúc và các thành phần chức năng

□ Bộ nhớ trong:

- Lưu trữ lệnh và dữ liệu để CPU xử lý

- Bao gồm:

- ROM – Read Only Memory:

- Lưu trữ lệnh và dữ liệu của hệ thống
 - Thông tin trong ROM vẫn tồn tại khi mất nguồn nuôi

- RAM – Random Access Memory:

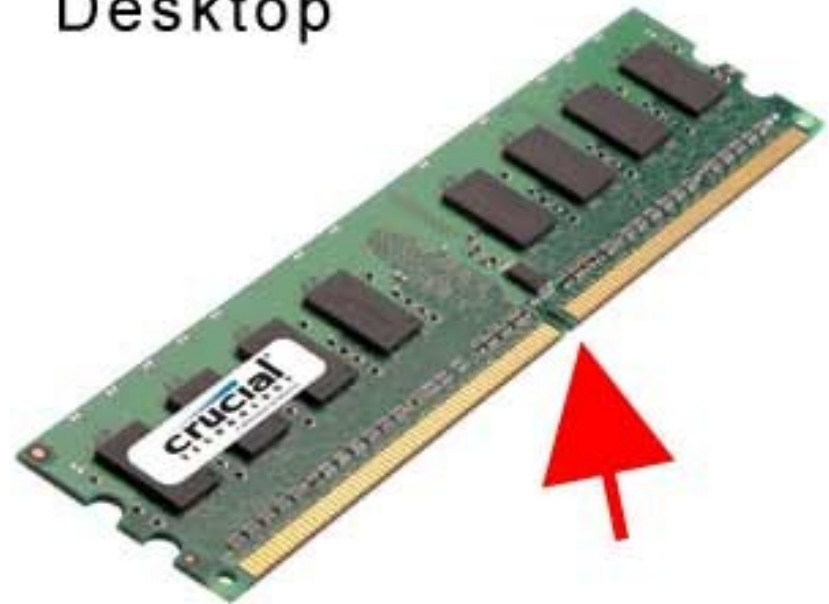
- Lưu trữ lệnh và dữ liệu của hệ thống và người dùng
 - Thông tin trong RAM sẽ mất khi mất nguồn nuôi

Các thành phần chính – bộ nhớ trong

ROM-BIOS



Crucial 240 pin DIMM
Desktop



2. Cấu trúc và các thành phần chức năng

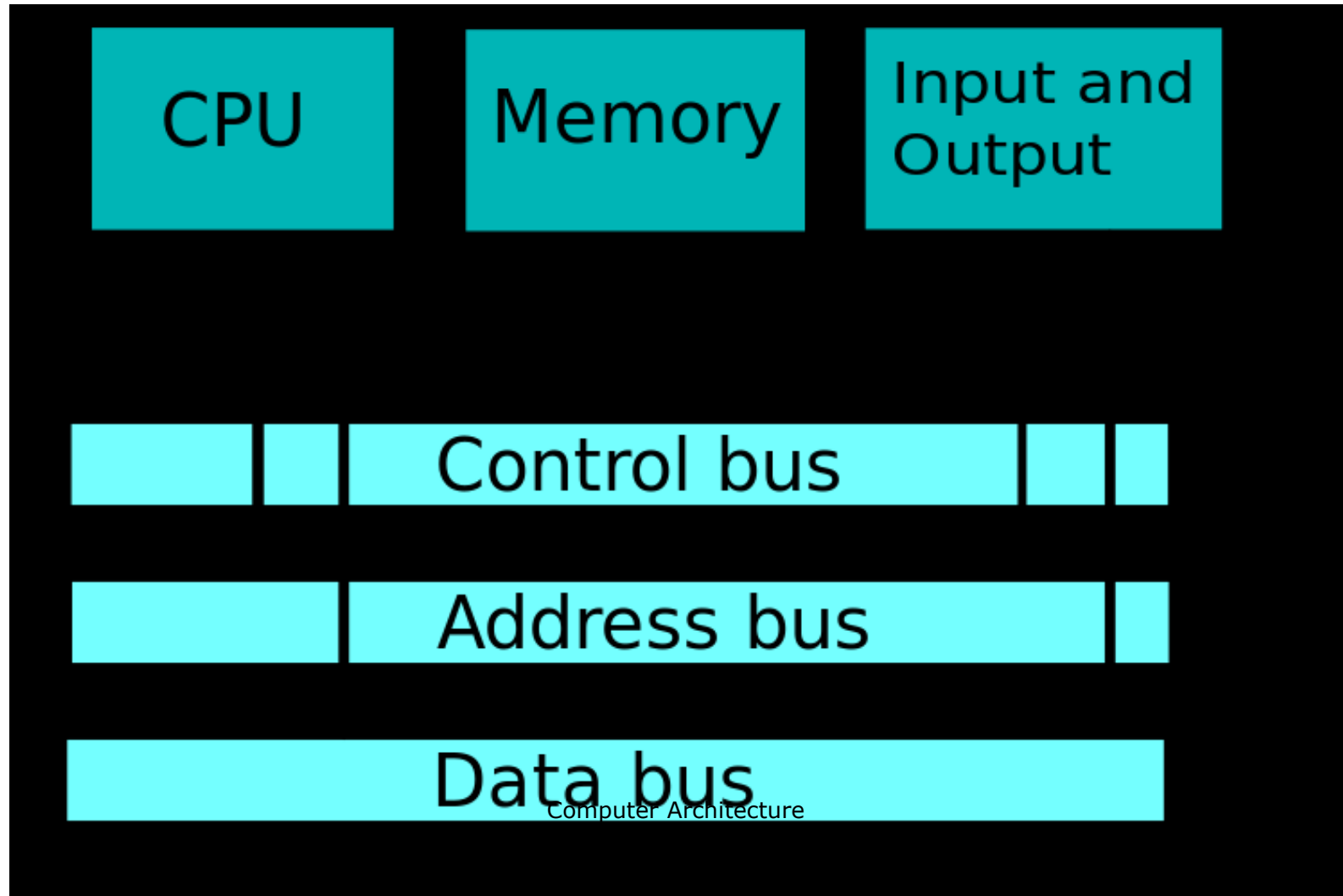
□ Các thiết bị vào ra:

- Thiết bị vào (input devices): nhập dữ liệu và điều khiển
 - Bàn phím
 - Chuột
 - Ổ đĩa
 - Máy quét
- Thiết bị ra: kết xuất dữ liệu
 - Màn hình
 - Máy in
 - Ổ đĩa

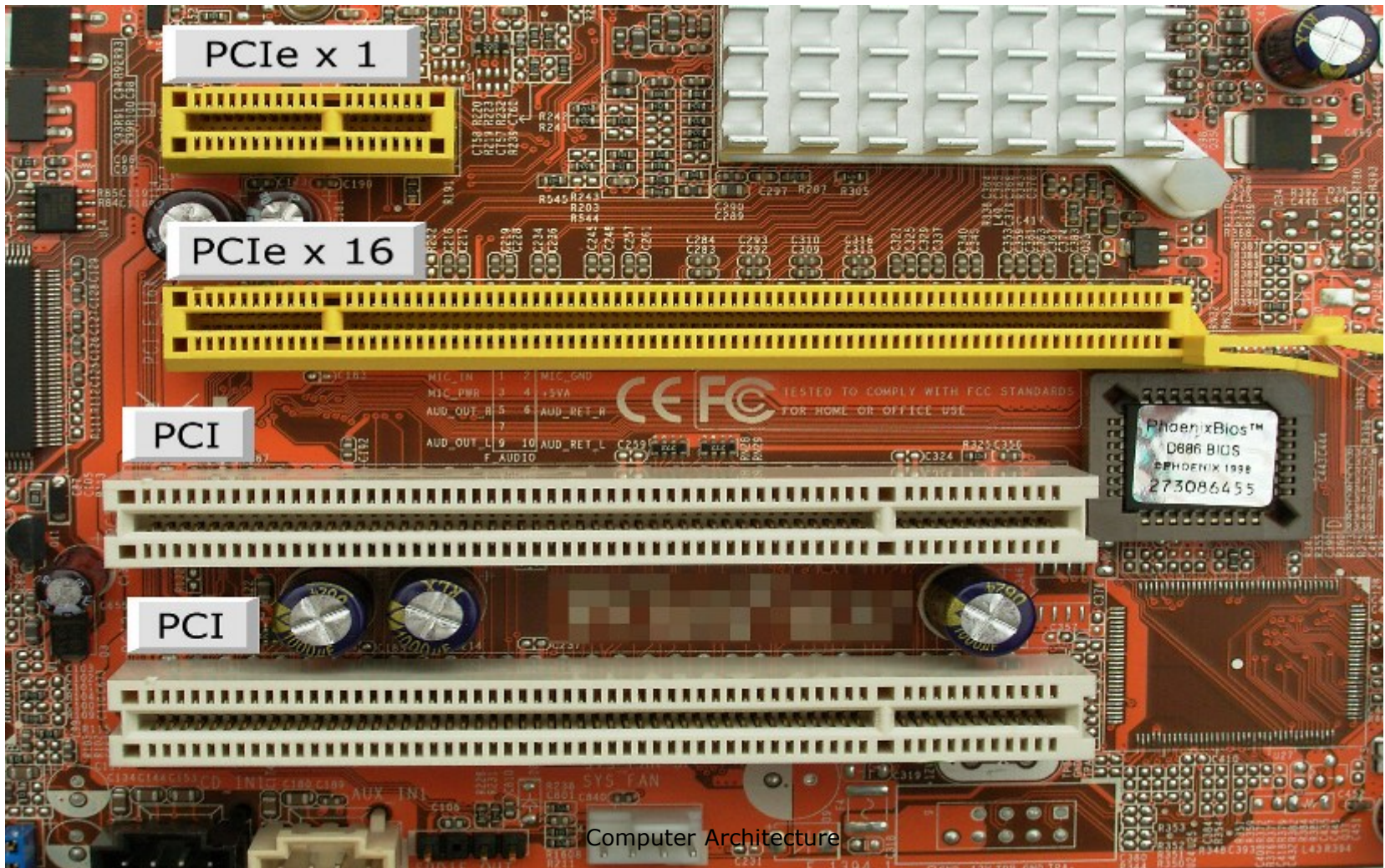
Các thành phần chính – bus hệ thống

- Tập các đường dây kết nối CPU với các thành phần khác của máy tính
- Bao gồm 3 loại:
 - ▣ Bus địa chỉ (gọi là bus A)
 - ▣ Bus dữ liệu (gọi là bus D)
 - ▣ Bus điều khiển (bus C)

Bus hệ thống



PCI bus

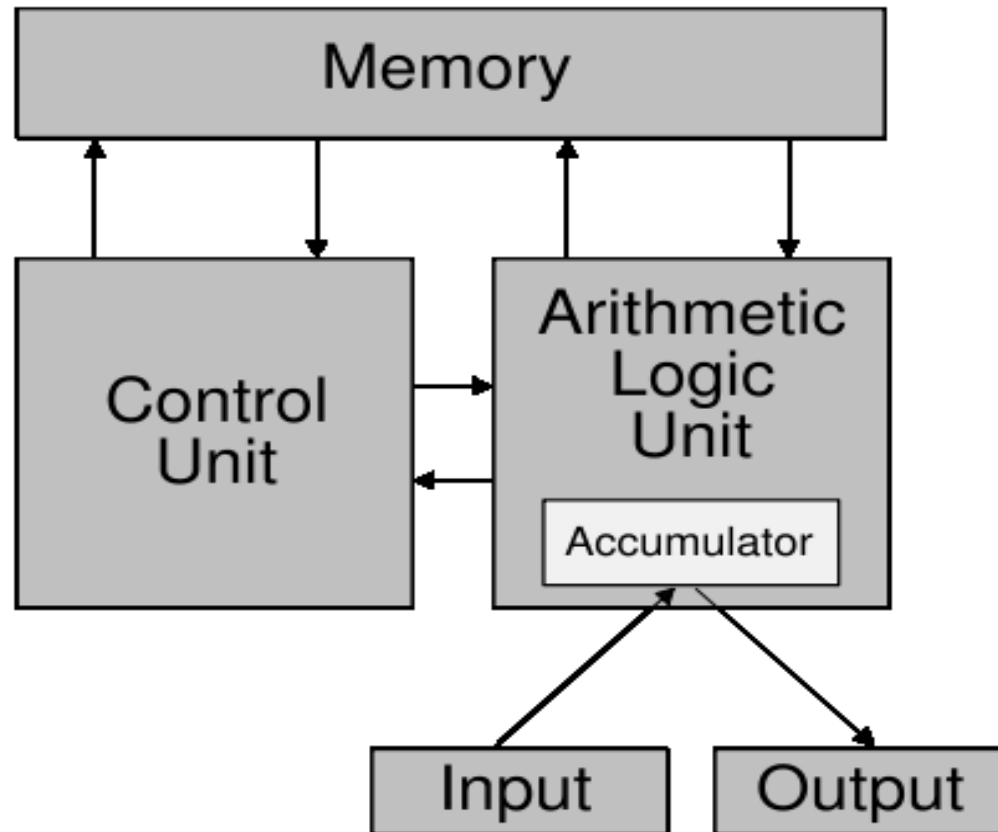


6. Thiết kế kiến trúc máy tính

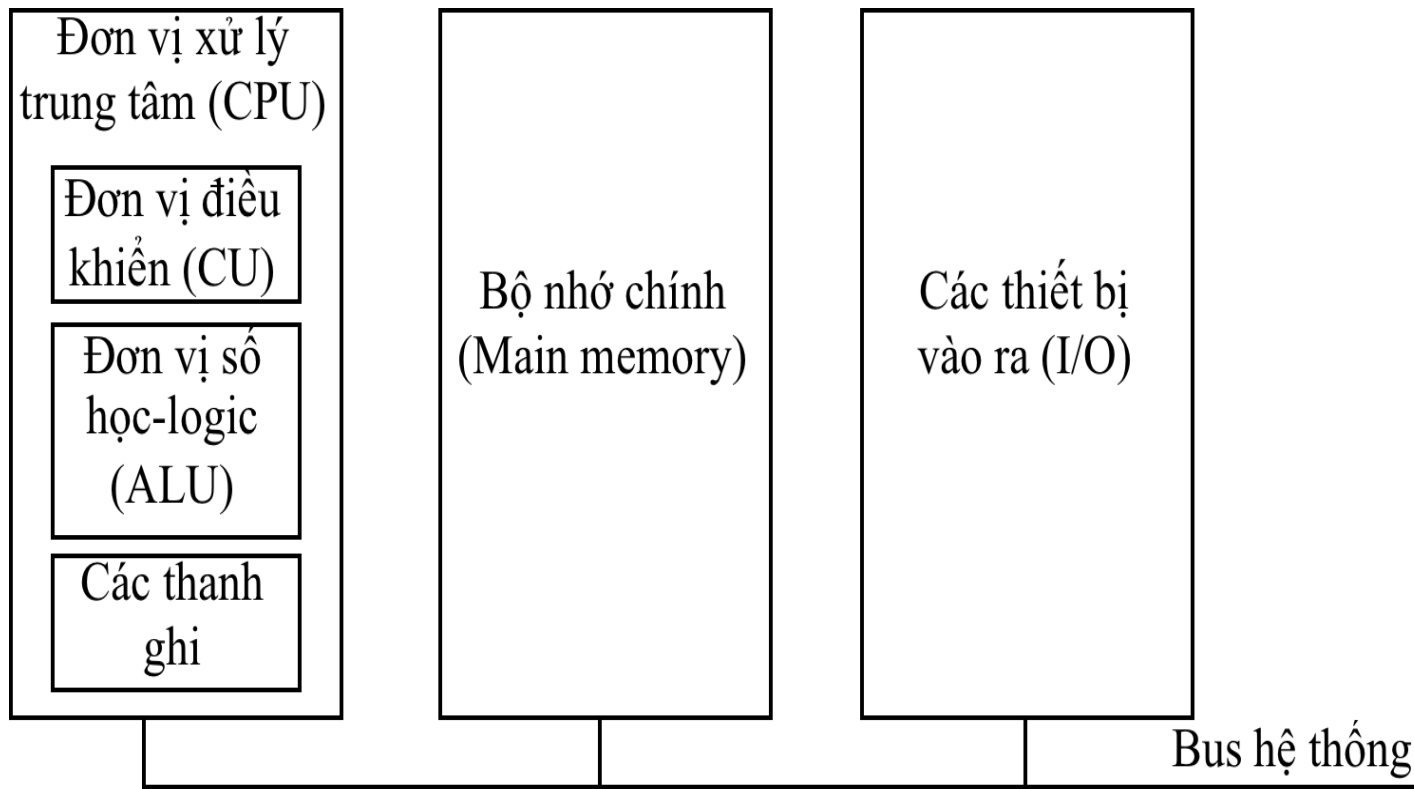
6. Thiết kế kiến trúc máy tính

- Theo sự phát triển, có nhiều loại kiến trúc ra đời -> nghiên cứu 2 kiến trúc chính sau
- **Kiến trúc Von-Neuman**
- **Kiến trúc Harvard**
- ... (>20 kiến trúc máy tính khác nhau)
- Von Neumann Architecture - Computer Science GCSE GURU

Kiến trúc Von-Neumann cũ



Kiến trúc Von-Neumann hiện đại



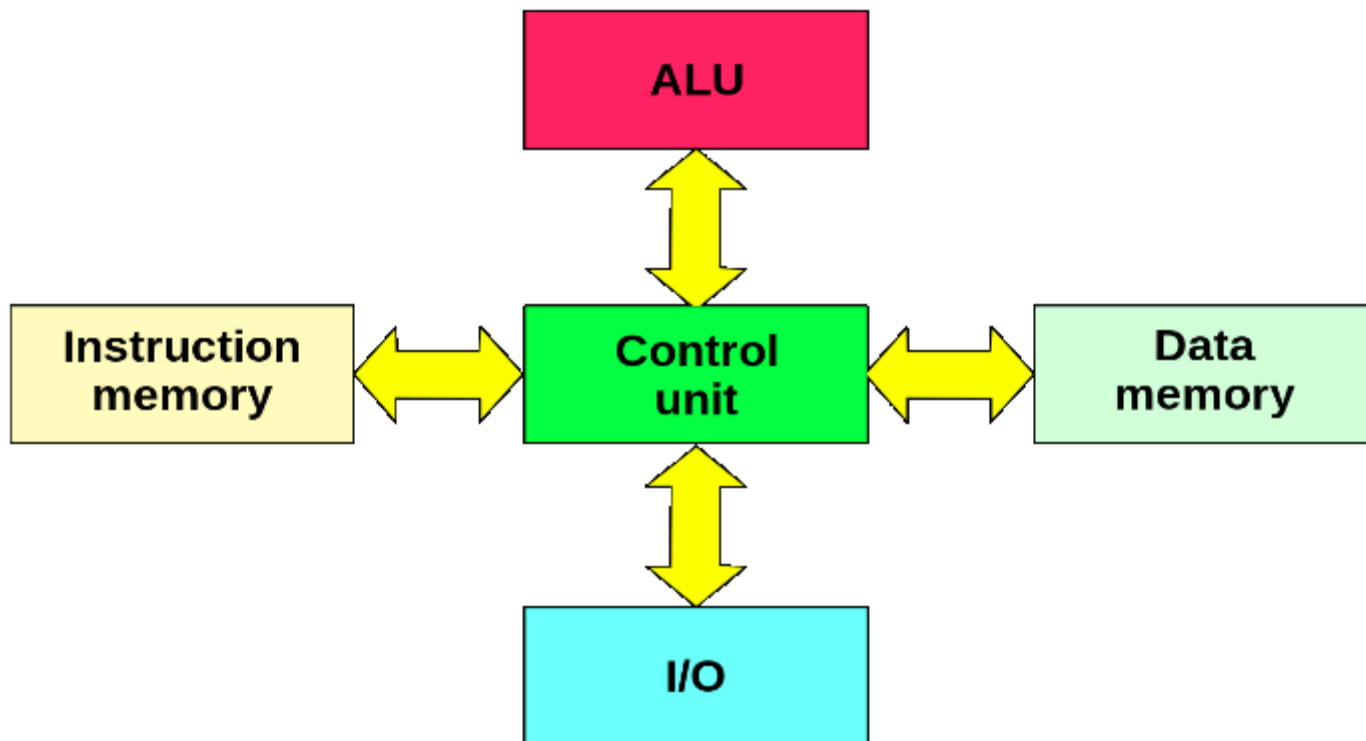
Kiến trúc Von-Neumann - Các đặc điểm

- ❑ Quá trình thực hiện lệnh được chia thành 3 giai đoạn chính :
 - CPU lấy lệnh (fetch) từ bộ nhớ
 - CPU giải mã lệnh và chạy lệnh; nếu lệnh cần dữ liệu thì đọc dữ liệu từ bộ nhớ
 - CPU viết kết quả vào bộ nhớ nếu có
- ❑ Hạn chế: bộ nhớ lệnh và dữ liệu (cổ chai) ko đc truy cập cùng lúc nên thông lượng (throughput) nhỏ hơn rất nhiều so với tốc độ CPU có thể làm việc
- ❑ Khắc phục: Dùng bộ nhớ cache giữa CPU và main memory

Kiến trúc Harvard

- Khắc phục được khuyết điểm của kiến trúc Von-Neumann
- Bộ nhớ được chia thành 2 phần:
 - Bộ nhớ chương trình
 - Bộ nhớ dữ liệu
- CPU sử dụng 2 bus hệ thống để liên hệ với bộ nhớ:
 - CPU có thể đọc lệnh và truy cập dữ liệu bộ nhớ cùng một lúc.
 - Một bus A,D cho bộ nhớ chương trình và 1 bus A,D cho bộ nhớ dữ liệu (khác nhau về định dạng)

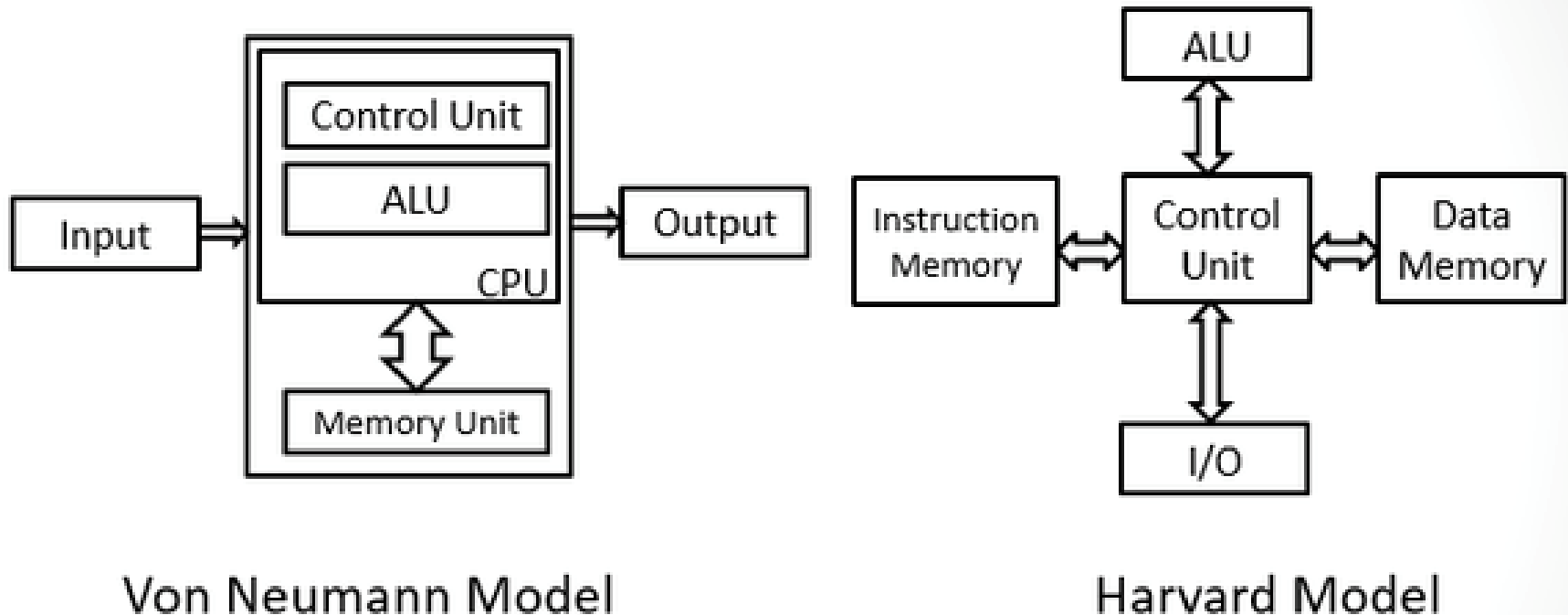
Kiến trúc Harvard



Kiến trúc Harvard

- Nhanh hơn vì băng thông bus rộng vì quá trình đọc lệnh không tranh chấp với quá trình truy xuất dữ liệu
- Hỗ trợ nhiều truy cập đọc/viết bộ nhớ cùng lúc → giảm xung đột truy cập bộ nhớ
- Ngày nay kiến trúc Harvard cải tiến được ứng dụng cho các kiến trúc máy tính hiện đại: ARM, intel x86
- Kiến trúc Harvard cũng được ứng dụng ở các hệ thống nhúng embedded system, chip chuyên xử lý tín hiệu (DSP)

So sánh Von Neumann và Harvard



Von Neumann: địa chỉ và bộ nhớ chia sẻ dữ liệu và lệnh từ CPU
Harvard: 2 địa chỉ và bộ nhớ cho dữ liệu và CPU → xử lý song song

So Sánh

Harvard Architecture	Von Neumann Architecture
Kiến trúc Harvard có các đường bus riêng biệt cho lệnh và dữ liệu.	Kiến trúc Von Neumann sử dụng chung một đường bus cho cả dữ liệu và lệnh.
Nó có một hệ thống bus địa chỉ và bus dữ liệu chuyên dụng để đọc dữ liệu từ và ghi dữ liệu vào bộ nhớ, và một bộ khác để tìm nạp các lệnh.	Sử dụng chung một hệ thống bus địa chỉ và dữ liệu để đọc ghi và tìm nạp lệnh
Theo kiến trúc harvard, CPU có thể vừa đọc lệnh vừa thực hiện truy cập bộ nhớ dữ liệu cùng một lúc.	CPU có thể đọc lệnh hoặc đọc / ghi dữ liệu từ / vào bộ nhớ. Cả hai không thể xảy ra cùng một lúc vì các hướng dẫn và dữ liệu sử dụng cùng một hệ thống xe buýt.
Máy kiến trúc Harvard có lệnh và không gian địa chỉ dữ liệu riêng biệt: địa chỉ lệnh số 0 không giống với địa chỉ dữ liệu số 0.	Máy kiến trúc Von Neumann có lệnh và không gian địa chỉ dữ liệu giống nhau

Biểu diễn dữ liệu trong máy tính

- Sử dụng hệ nhị phân để biểu diễn dữ liệu
- Hệ nhị phân sử dụng 2 kí tự 0 và 1; 0 biểu diễn giá trị logic False; 1 biểu diễn giá trị logic True.
- Hệ hexa cũng được sử dụng; gồm 16 kí tự: 0-9, A, B, C, D, E, F.

Các hệ đếm cơ bản

- Hệ thập phân (Decimal System)
→ con người sử dụng
- Hệ nhị phân (Binary System)
→ máy tính sử dụng
- Hệ mười sáu (Hexadecimal System)
→ dùng để viết gọn cho số nhị phân

1. Hệ thập phân

- Cơ số 10
- 10 chữ số: 0,1,2,3,4,5,6,7,8,9
- Dùng n chữ số thập phân có thể biểu diễn được 10^n giá trị khác nhau:
 - $00\dots000 = 0$
 - $99\dots999 = 10^n - 1$

Dạng tổng quát của số thập phân

$$A = a_n a_{n-1} \dots a_1 a_0, a_{-1} \dots a_{-m}$$

Giá trị của A được hiểu như sau:


$$A = a_n 10^n + a_{n-1} 10^{n-1} + \dots + a_1 10^1 + a_0 10^0 + a_{-1} 10^{-1} + \dots + a_{-m} 10^{-m}$$

$$A = \sum_{i=-m}^n a_i 10^i$$


Ví dụ số thập phân

$$472.38 = 4 \times 10^2 + 7 \times 10^1 + 2 \times 10^0 + 3 \times 10^{-1} + 8 \times 10^{-2}$$

■ Các chữ số của phần nguyên:

- $472 : 10 = 47$ dư 2
 - $47 : 10 = 4$ dư 7
 - $4 : 10 = 0$ dư 4
- 

■ Các chữ số của phần lẻ:

- $0.38 \times 10 = 3.8$ phần nguyên = 3
 - $0.8 \times 10 = 8.0$ phần nguyên = 8
- 

2. Hệ nhị phân

- Cơ số 2
- 2 chữ số nhị phân: 0 và 1 (**binary digit**)
- chữ số nhị phân gọi là **bit**
- Bit là đơn vị thông tin nhỏ nhất
- Dùng n bit có thể biểu diễn được 2^n giá trị khác nhau:
 - $00\dots000 = 0$
 - $11\dots111 = 2^n - 1$

Bits, Bytes, Nibbles...

■ Bits

10010110
└─┬─┘ └─┬─┘
most least
significant significant
bit bit

■ Bytes & Nibbles

byte
┌──────────┐
10010110
└────────┘
nibble

■ Bytes

CEBF9AD7
└─┬─┘ └─┬─┘
most least
significant significant
byte byte

Lũy thừa hai

- $2^{10} = 1 \text{ Ki}$ $\approx 1000 \text{ (1024)}$
- $2^{20} = 1 \text{ Mi}$ $\approx 1 \text{ triệu (1,048,576)}$
- $2^{30} = 1 \text{ Gi}$ $\approx 1 \text{ tỷ (1,073,741,824)}$
- $2^{40} = 1 \text{ Ti}$ $\approx 1000 \text{ tỷ}$
- $2^{50} = 1 \text{ Pi}$ $\approx 1 \text{ triệu tỷ}$

Dạng tổng quát của số nhị phân

Có một số nhị phân A như sau:

$$A = a_n a_{n-1} \dots a_1 a_0, a_{-1} \dots a_{-m}$$

Giá trị của A được tính như sau:

$$A = a_n 2^n + a_{n-1} 2^{n-1} + \dots + a_1 2^1 + a_0 2^0 + a_{-1} 2^{-1} + \dots + a_{-m} 2^{-m}$$

$$A = \sum_{i=-m}^n a_i 2^i$$

Ví dụ số nhị phân

$$\underset{\substack{6\ 5\ 4\ 3\ 2\ 1\ 0\ -1\ -2\ -3\ -4}}{1101001.1011}_{(2)} =$$

$$= 1x2^6 + 1x2^5 + 0x2^4 + 1x2^3 + 0x2^2 + 0x2^1 + 1x2^0 + 1x2^{-1} + 0x2^{-2} + 1x2^{-3} + 1x2^{-4}$$

$$= 2^6 + 2^5 + 2^3 + 2^0 + 2^{-1} + 2^{-3} + 2^{-4}$$

$$= 64 + 32 + 8 + 1 + 0.5 + 0.125 + 0.0625$$

$$= 105.6875_{(10)}$$

Chuyển đổi số nguyên thập phân sang nhị phân

- Phương pháp 1: chia dần cho 2 rồi lấy phần dư
- Phương pháp 2: Phân tích thành tổng của các số 2^i → nhanh hơn

Phương pháp chia dần cho 2

Ví dụ: chuyển đổi

105,6875₍₁₀₎

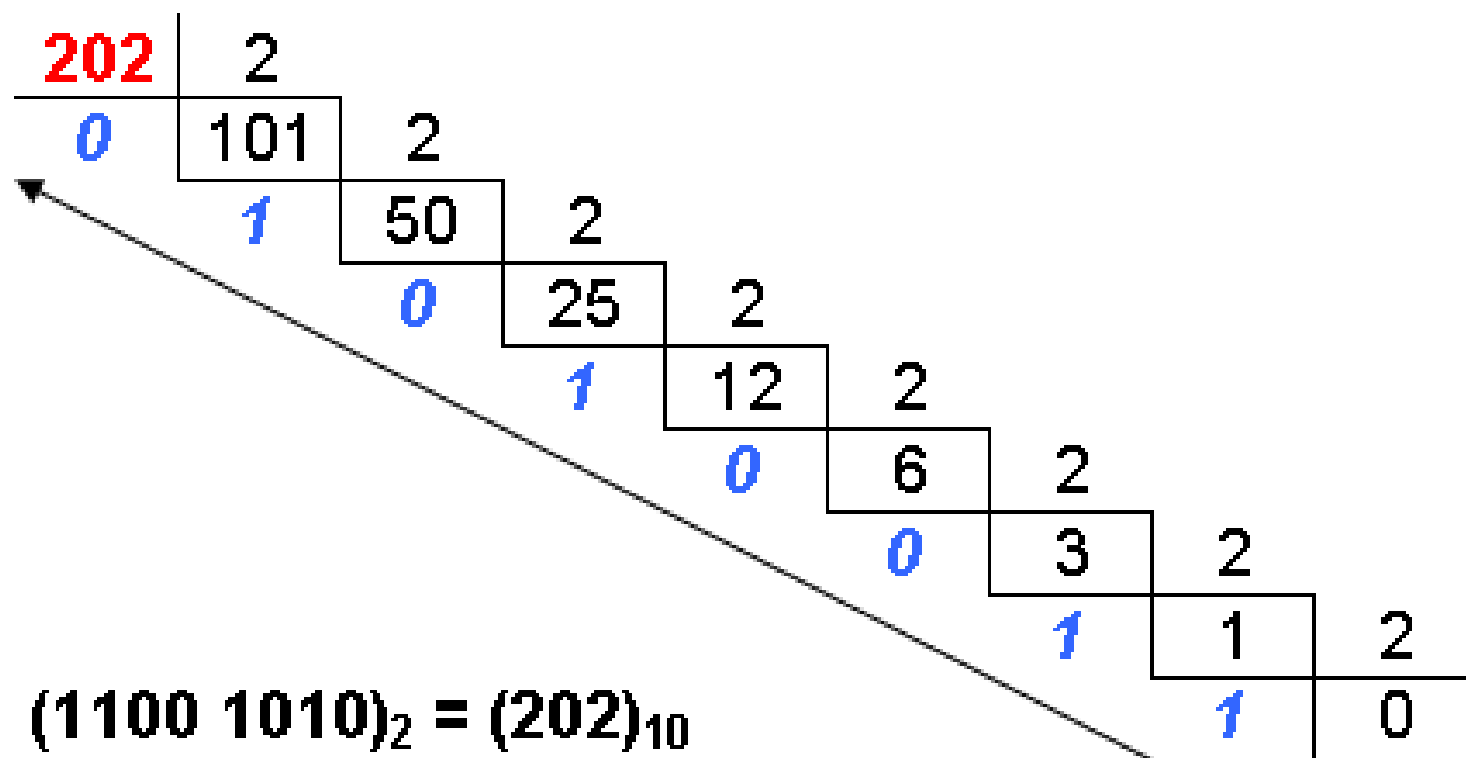
105 : 2 = 52	dư 1
52 : 2 = 26	dư 0
26 : 2 = 13	dư 0
13 : 2 = 6	dư 1
6 : 2 = 3	dư 0
3 : 2 = 1	dư 1
1 : 2 = 0	dư 1

0.6875 x 2 = 1.375	phần nguyên = 1
0.375 x 2 = 0.75	phần nguyên = 0
0.75 x 2 = 1.5	phần nguyên = 1
0.5 x 2 = 1.0	phần nguyên = 1

Kết quả:

105₍₁₀₎ = 1101001, 1011₍₂₎

Chuyển đổi số thập phân sang nhị phân



Phương pháp phân tích thành tổng của các 2^i

- Ví dụ 1: chuyển đổi $105,6875_{(10)}$
 - $105,6875 = 64 + 32 + 8 + 1 + 0.5 + 0.125 + 0.0625$
 $= 2^6 + 2^5 + 2^3 + 2^0 + 2^{-1} + 2^{-3} + 2^{-4}$

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
128	64	32	16	8	4	2	1	0.5	0.25	0.125	0.0375
0	1	1	0	1	0	0	1	1	0	1	1

- Kết quả: $105_{(10)} = 011010011011_{(2)}$

Chuyển đổi số thập phân sang nhị phân

- Chuyển $(0.6875)_{10} \rightarrow (.)_2$?
- Chuyển $(0.81)_{10} \rightarrow (.)_2$?
- Chuyển $(0.2)_{10}$?

3. Hệ mười sáu (Hexa)

- Cơ số 16
- 16 chữ số: 0,1,2,3,4,5,6,7,8,9, A,B,C,D,E,F
- Dùng để viết gọn cho số nhị phân: cứ một nhóm 4-bit sẽ được thay bằng một chữ số Hexa

Hệ Hexa

- Mỗi kí hiệu trong hệ hexa được biểu diễn bởi 4 kí hiệu trong hệ nhị phân

Hexa	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Binary	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

Quan hệ giữa số nhị phân và số Hexa

4-bit	Chữ số Hexa
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Ví dụ chuyển đổi số nhị phân \rightarrow số Hexa:

- $1011\ 0011_2 = B3_{16}$

- $0000\ 0000_2 = 00_{16}$

- $0010\ 1101\ 1001\ 1010_2 = 2D9A_{16}$

- $1111\ 1111\ 1111\ 1111_2 = FFFF_{16}$

Tổ chức dữ liệu

□ Bits:

- Là đơn vị dữ liệu nhỏ nhất
- Một bit chỉ có thể lưu trữ 2 giá trị: 0 hoặc 1, true hoặc false.

□ Nibbles:

- Nhóm 4 bits
- Có thể lưu trữ tới 16 giá trị từ $(0000)_2$ tới $(1111)_2$, hoặc 1 số hệ hexa.

Data Organization

□ Bytes:

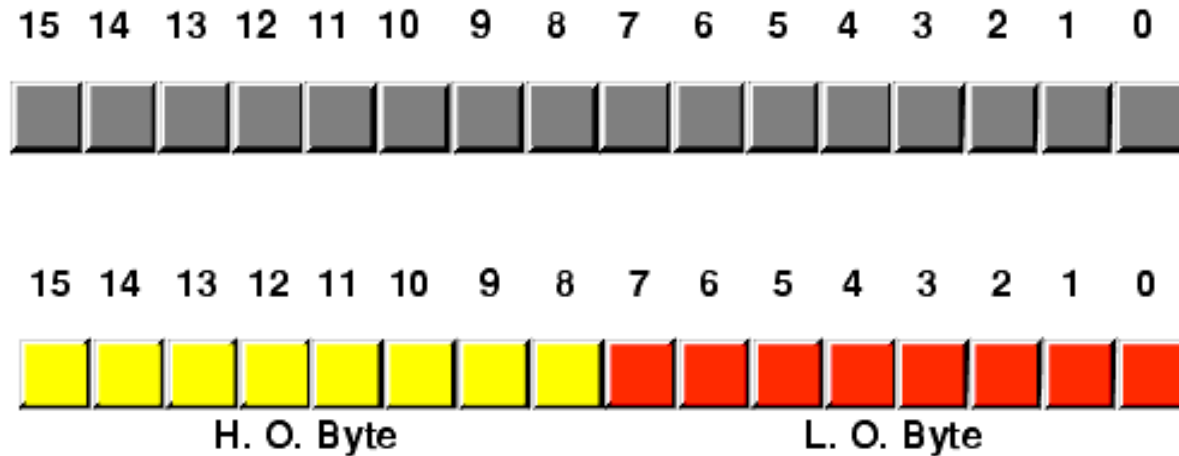
- Nhóm 8 bit hoặc 2 nibbles
- Có thể lưu tới 256 giá trị, từ $(0000\ 0000)_2$ tới $(1111\ 1111)_2$, hoặc từ $(00)_{16}$ tới $(FF)_{16}$.



Data Organization

□ Words (từ):

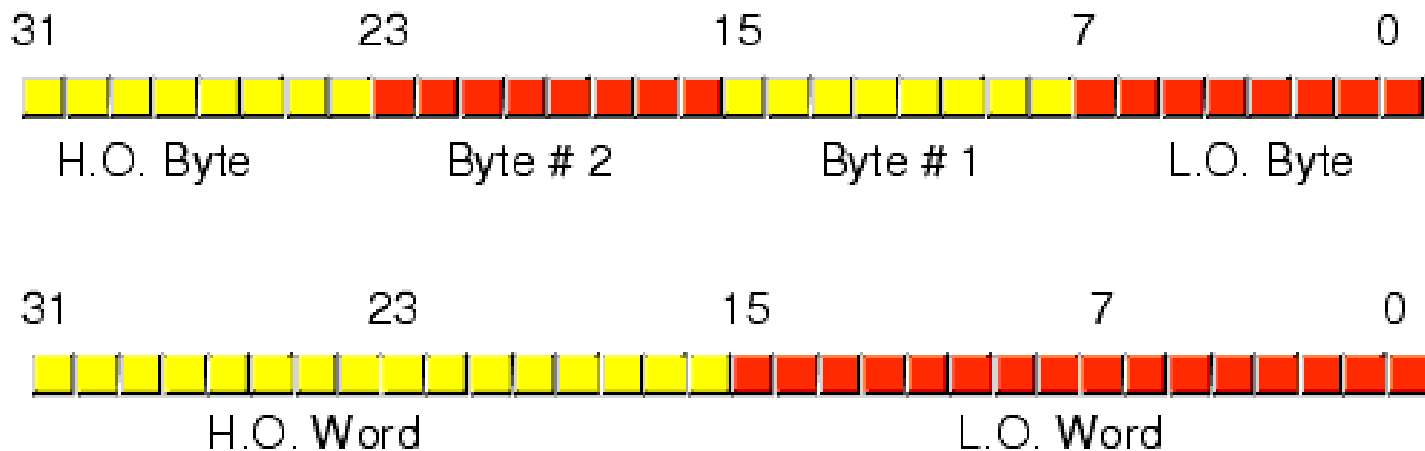
- Nhóm 16 bits, hay 2 bytes
- Có thể lưu tới 2^{16} (65536) values, từ $(0000)_{16}$ tới $(FFFF)_{16}$.



Data Organization

□ Double words:

- A double word nhóm 32 bits, hoặc 4 bytes, hoặc 2 words
- Có thể lưu tới 2^{32} values, từ $(0000\ 0000)_{16}$ tới $(FFFF\ FFFF)_{16}$.



Số có dấu và không dấu

- Trong hệ nhị phân, bit trái nhất được dùng để biểu diễn dấu của số có dấu
 - Bit trái nhất là 0 \rightarrow số dương
 - Bit trái nhất là 1 \rightarrow số âm
- Ví dụ: sử dụng 4 bit để biểu diễn các số
 - 0011, 0111, 0101 là các số dương
 - 1011, 1111, 1101 là các số âm
- Đối với các số không dấu, tất cả các bit đều lưu giá trị

Số có dấu và không dấu

- Phạm vi biểu diễn: n bits có thể biểu diễn:
 - Số có dấu: từ -2^{n-1} tới $+2^{n-1}-1$
 - 8 bits: từ -128 tới +127
 - 16 bits: từ -32768 tới +32767
 - 32 bits: từ -2,147,483,648 tới +2,147,483,647
 - Số không dấu: từ 0 tới 2^n
 - 8 bits: từ 0 tới 256
 - 16 bits: từ 0 tới 65536
 - 32 bits: từ 0 tới 4,294,967,296

Bảng mã ASCII

- ❑ ASCII (American Standard Code for Information Interchange) là bảng mã các kí tự chuẩn tiếng Anh dùng cho trao đổi dữ liệu trong các hệ thống tính toán.
- ❑ Sử dụng 8 bit để biểu diễn 1 kí tự.
- ❑ Mã ASCII gồm định nghĩa cho 128 kí tự :
 - 33 kí tự điều khiển
 - 94 kí tự
- ❑ Các giá trị còn lại (129-255) dự trữ

ASCII Table – Control chars

Binary	Oct	Dec	Hex	Abbr	PR ^[t 1]	CS ^[t 2]	CEC ^[t 3]	Description
000 0000	000	0	00	NUL	NUL	^@	\0	Null character
000 0001	001	1	01	SOH	SOH	^A		Start of Header
000 0010	002	2	02	STX	STX	^B		Start of Text
000 0011	003	3	03	ETX	ETX	^C		End of Text
000 0100	004	4	04	EOT	EOT	^D		End of Transmission
000 0101	005	5	05	ENQ	ENQ	^E		Enquiry
000 0110	006	6	06	ACK	ACK	^F		Acknowledgment
000 0111	007	7	07	BEL	BEL	^G	\a	Bell
000 1000	010	8	08	BS	BS	^H	\b	Backspace ^{[t 4][t 5]}
000 1001	011	9	09	HT	HT	^I	\t	Horizontal Tab
000 1010	012	10	0A	LF	LF	^J	\n	Line feed

ASCII Table – Printable chars

Binary	Oct	Dec	Hex	Glyph
010 0000	040	32	20	SP
010 0001	041	33	21	!
010 0010	042	34	22	"
010 0011	043	35	23	#
010 0100	044	36	24	\$
010 0101	045	37	25	%
010 0110	046	38	26	&
010 0111	047	39	27	'
010 1000	050	40	28	(
010 1001	051	41	29)
010 1010	052	42	2A	*
010 1011	053	43	2B	+
010 1100	054	44	2C	,

Binary	Oct	Dec	Hex	Glyph
100 0000	100	64	40	@
100 0001	101	65	41	A
100 0010	102	66	42	B
100 0011	103	67	43	C
100 0100	104	68	44	D
100 0101	105	69	45	E
100 0110	106	70	46	F
100 0111	107	71	47	G
100 1000	110	72	48	H
100 1001	111	73	49	I
100 1010	112	74	4A	J
100 1011	113	75	4B	K
100 1100	114	76	4C	L

Binary	Oct	Dec	Hex	Glyph
110 0000	140	96	60	`
110 0001	141	97	61	a
110 0010	142	98	62	b
110 0011	143	99	63	c
110 0100	144	100	64	d
110 0101	145	101	65	e
110 0110	146	102	66	f
110 0111	147	103	67	g
110 1000	150	104	68	h
110 1001	151	105	69	i
110 1010	152	106	6A	j
110 1011	153	107	6B	k
110 1100	154	108	6C	l