

Chương 3: Hệ thống nhớ và Bộ nhớ Cache

Kiến trúc máy tính

ThS. Đinh Xuân Trường
truongdx@ptit.edu.vn



Posts and Telecommunications
Institute of Technology
Faculty of Information Technology 1



CNTT1
Học viện Công nghệ Bưu chính Viễn thông

January 15, 2023

Mục tiêu Buổi 6

Mô hình phân cấp bộ nhớ

Phân loại bộ nhớ và tổ chức mạch nhớ

Bộ nhớ chính

ROM

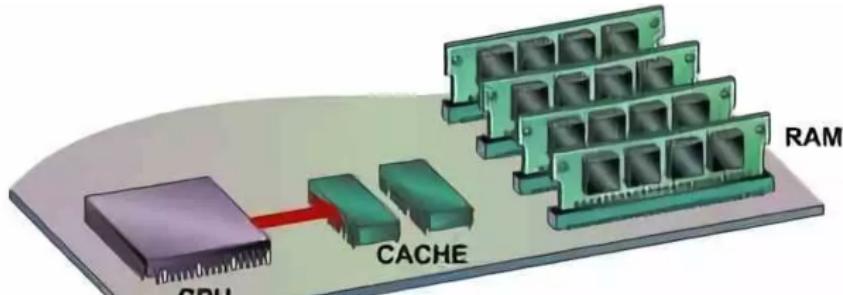
RAM

Bộ nhớ Cache

Bộ nhớ Cache là gì?

Kiến trúc và tổ chức Bộ nhớ Cache

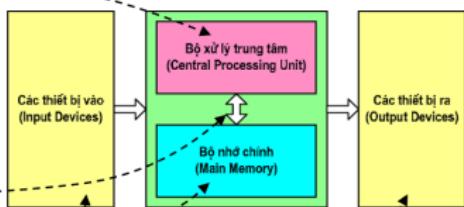
Cách thức để nâng cao hiệu năng của Cache



Các đặc trưng của bộ nhớ máy tính

Vị trí

- Bên trong CPU:
 - **Tập thanh ghi**
- Bộ nhớ trong:
 - Bộ nhớ cache
 - **Bộ nhớ chính**
- **Bộ nhớ ngoài: các thiết bị nhớ**



Dung lượng

- Độ dài từ nhớ - tính bằng bit
- Số lượng từ nhớ



Mô hình phân cấp bộ nhớ (cont.)

Các đặc trưng của bộ nhớ máy tính

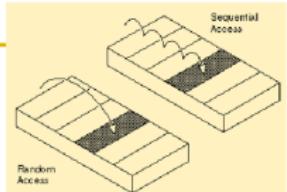
Đơn vị truyền

- Từ nhớ - Word
- Khối nhớ - Block



Phương pháp truy nhập

- Truy nhập tuần tự (băng từ)
- Truy nhập trực tiếp (các loại đĩa)
- Truy nhập ngẫu nhiên (bộ nhớ bán dẫn)
- Truy nhập liên kết (cache)



Các đặc trưng của bộ nhớ máy tính

Đặc tính vật lý

- Khả biến / Không khả biến (volatile / nonvolatile)
- Xoá được / Không xoá được

Kiểu vật lý

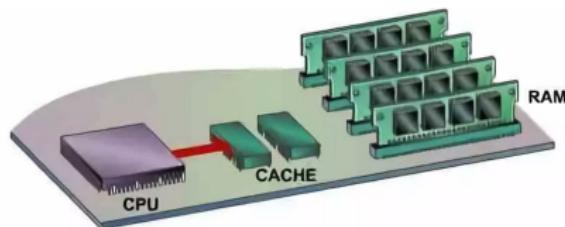
- Bộ nhớ bán dẫn
- Bộ nhớ từ
- Bộ nhớ quang

► Thanh ghi của CPU:

- Kích thước rất nhỏ (vài chục byte tới vài KB)
- Tốc độ rất nhanh, thời gian truy cập khoảng 0.25 ns, giá thành đắt
- Lưu trữ tạm thời dữ liệu đầu vào và ra cho các lệnh

► Cache:

- Kích thước nhỏ (64KB tới 16MB)
- Tốc độ nhanh, thời gian truy cập khoảng 1 – 5ns, giá thành đắt
- Lưu trữ lệnh và dữ liệu cho CPU
- Còn được gọi là "*bộ nhớ thông minh*"(smart memory)



Mô hình phân cấp bộ nhớ (cont.)

Các thành phần phân cấp bộ nhớ

► Bộ nhớ chính:

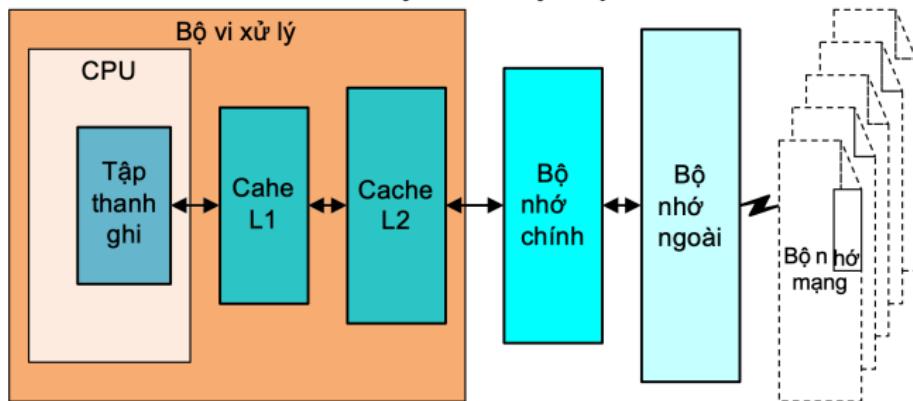
- Kích thước lớn, dung lượng từ 256MB tới 4GB cho các hệ 32bits
- Tốc độ chậm, thời gian truy cập từ 50 – 70ns
- Lưu trữ lệnh và dữ liệu cho hệ thống và người dùng
- Giá thành rẻ



► Bộ nhớ phụ:

- Kích thước rất lớn, dung lượng từ 20GB tới 1000GB
- Tốc độ rất chậm, thời gian truy cập khoảng 5ms
- Lưu trữ lượng dữ liệu lớn dưới dạng file trong thời gian lâu dài
- Giá thành rất rẻ

Cấu trúc phân cấp bộ nhớ



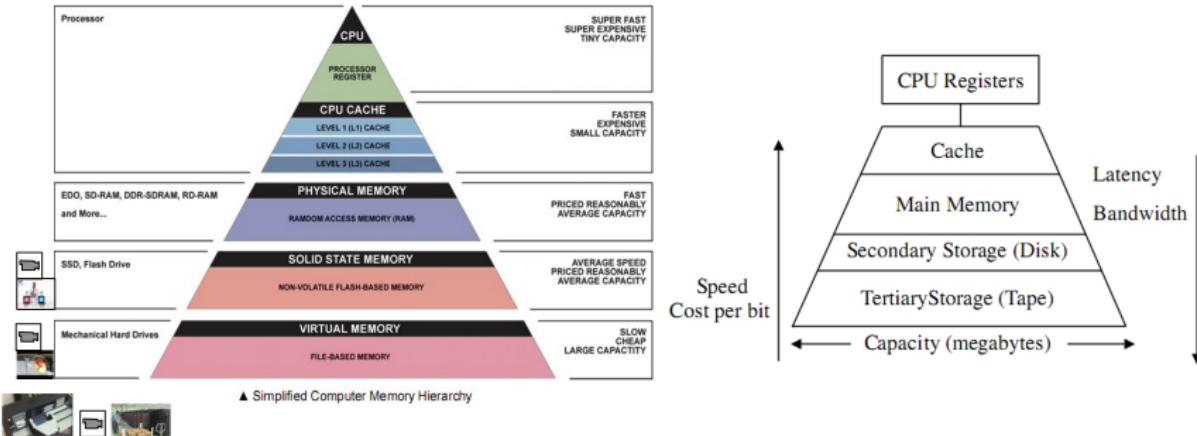
Từ trái sang phải:

- ▶ Dung lượng tăng dần
- ▶ Tốc độ giảm dần
- ▶ Giá thành / 1 bit tăng dần

Mô hình phân cấp bộ nhớ (cont.)

Vai trò của mô hình phân cấp

Cấu trúc phân cấp bộ nhớ



Các tham số phân cấp bộ nhớ

	Access type	Capacity	Latency	Bandwidth	Cost/MB
CPU registers	Random	64–1024 bytes	1–10 ns	System clock rate	High
Cache memory	Random	8–512 KB	15–20 ns	10–20 MB/s	\$500
Main memory	Random	16–512 MB	30–50 ns	1–2 MB/s	\$20–50
Disk memory	Direct	1–20 GB	10–30 ms	1–2 MB/s	\$0.25
Tape memory	Sequential	1–20 TB	30–10,000 ms	1–2 MB/s	\$0.025

Vai trò của mô hình phân cấp

► Nâng cao hiệu năng hệ thống:

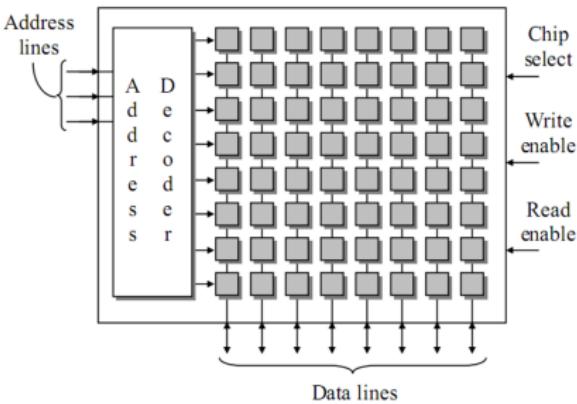
- Dung hòa được CPU có tốc độ cao với bộ nhớ chính và bộ nhớ phụ có tốc độ thấp
- Thời gian truy cập dữ liệu trung bình của CPU từ hệ thống bộ nhớ gần bằng thời gian truy cập cache



► Giảm giá thành sản xuất:

- Các thành phần đắt tiền sẽ được sử dụng với dung lượng nhỏ hơn
- Các thành phần rẻ hơn được sử dụng với dung lượng lớn hơn
- Tổng giá thành của hệ thống nhớ theo mô hình phân cấp sẽ rẻ hơn so với hệ thống nhớ không phân cấp cùng tốc độ

- ▶ Dựa vào kiểu truy cập:
 - Bộ nhớ truy cập ngẫu nhiên (RAM: Random Access Memory)
 - Bộ nhớ truy cập tuần tự (SAM: Serial Access Memory)
 - Bộ nhớ chỉ đọc (ROM: Read Only Memory)
- ▶ Dựa vào khả năng chịu đựng/ lưu giữ thông tin:
 - Bộ nhớ không ổn định (volatile memory): thông tin lưu trữ bị mất khi tắt nguồn
 - Bộ nhớ ổn định: thông tin lưu trữ được giữ lại khi tắt nguồn
- ▶ Dựa vào công nghệ chế tạo:
 - Bộ nhớ bán dẫn: ROM, RAM, SSD, USB, ...
 - Bộ nhớ từ: HDD, FDD, tape
 - Bộ nhớ quang: CD, DVD



▶ Address lines:

- Các đường địa chỉ nối tới bus A
- Truyền tín hiệu địa chỉ từ CPU tới mạch nhớ

▶ Address decoder:

- Bộ giải mã địa chỉ
- Dùng địa chỉ để chọn và kích hoạt ô nhớ/dòng nhớ cần truy nhập

► Data lines:

- Các đường dữ liệu kết nối với bus D
- Truyền dữ liệu từ bộ nhớ về CPU và ngược lại

► Chip select CS:

- Chân tín hiệu chọn chip
- Chip nhớ được kích hoạt khi $CS=0$. Thông thường CPU chỉ làm việc với 1 chip nhớ tại 1 thời điểm

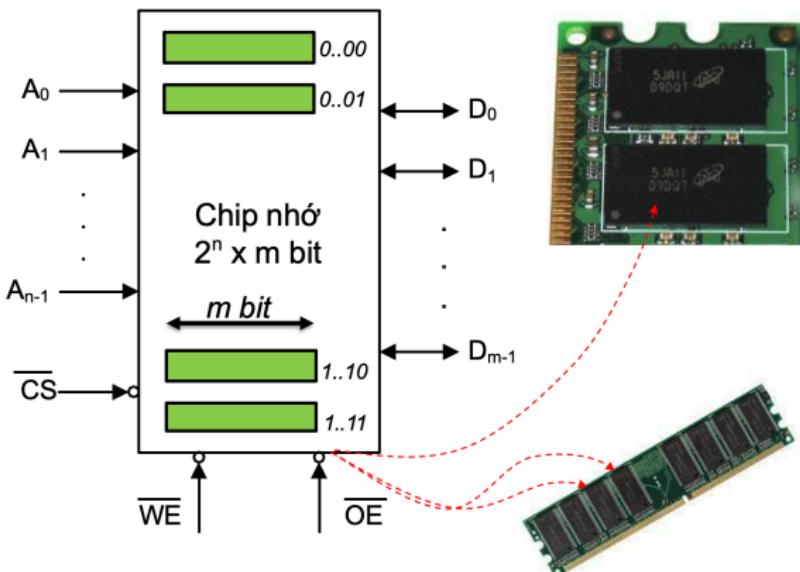
► Write enable WE:

- Chân tín hiệu cho phép ghi
- Cho phép ghi vào đường nhớ khi $WE = 0$

► Read enable RE:

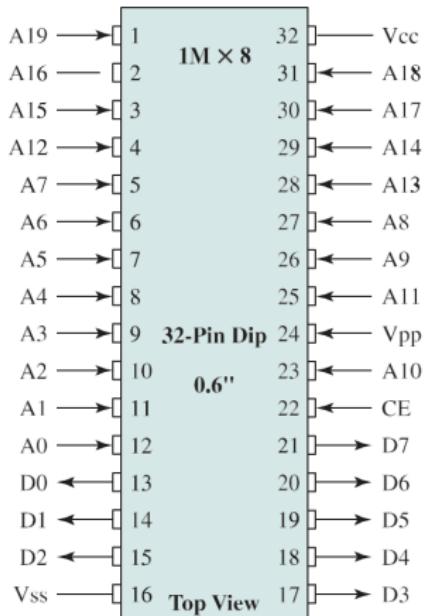
- Chân tín hiệu cho phép đọc
- Cho phép đọc dữ liệu từ đường nhớ khi $RE = 0$

Sơ đồ cơ bản của chip nhớ:

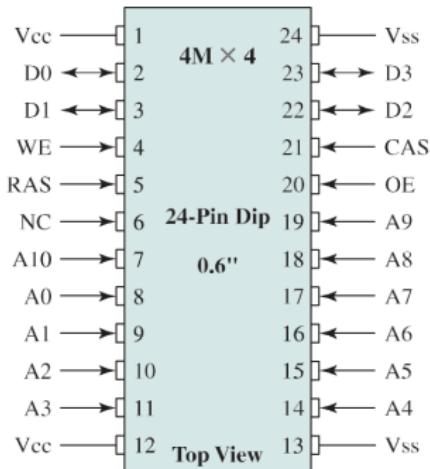


- ▶ Các đường địa chỉ: $A_{n-1} - A_0 \implies$ có 2^n từ nhớ
- ▶ Các đường dữ liệu: $D_{m-1} - D_0 \implies$ độ dài từ nhớ = m bit
- ▶ Dung lượng chip nhớ = $2^n \times m$ bit

Ví dụ về chíp nhớ:



(a) 8-Mbit EPROM



(b) 16-Mbit DRAM

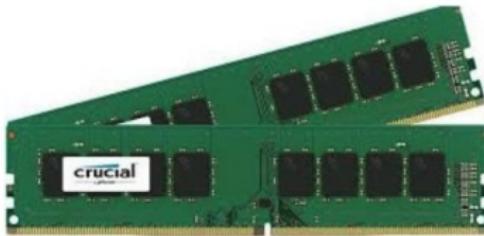
Bộ nhớ bán dẫn:

Kiểu bộ nhớ	Tiêu chuẩn	Khả năng xoá	Cơ chế ghi	Tính khả biến
Read Only Memory (ROM)	Bộ nhớ chỉ đọc	Không xoá được	Mặt nạ	Không khả biến
Programmable ROM (PROM)				
Erasable PROM (EPROM)	Bộ nhớ hầu như chỉ đọc	Băng tia cực tím, cả chip	Băng điện	Không khả biến
Electrically Erasable PROM (EEPROM)		Băng điện, mức từng byte		
Flash memory	Bộ nhớ đọc - ghi	Băng điện, từng khối	Băng điện	Khả biến
Random Access Memory (RAM)		Băng điện, mức từng byte		

Bộ nhớ chính (cont.)

Đặc trưng cơ bản của bộ nhớ trong:

- ▶ Chứa các chương trình đang thực hiện và chứa các dữ liệu đang được sử dụng
- ▶ Tồn tại trên mọi hệ thống máy tính
- ▶ Bao gồm các ngăn nhớ được đánh địa chỉ trực tiếp bởi CPU
- ▶ Dung lượng của bộ nhớ chính nhỏ hơn không gian địa chỉ bộ nhớ mà CPU quản lý.
- ▶ Việc quản lý logic bộ nhớ chính tuỳ thuộc vào hệ điều hành

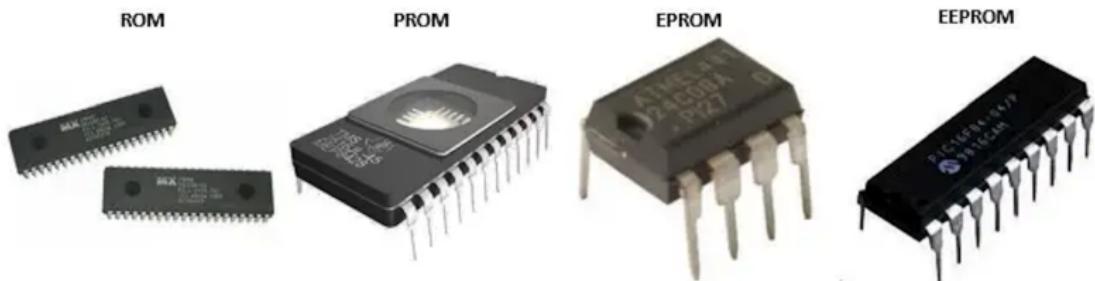


- ▶ ROM (Read Only Memory): là bộ nhớ chỉ đọc
 - Ghi thông tin vào ROM bằng cách sử dụng các thiết bị hoặc phương pháp đặc biệt
- ▶ ROM là bộ nhớ ổn định: tất cả thông tin vẫn được duy trì khi mất nguồn nuôi
- ▶ Là bộ nhớ bán dẫn: mỗi ô nhớ là một cổng bán dẫn
- ▶ Thường dùng để lưu trữ thông tin hệ thống: thông tin phần cứng và BIOS



Các loại ROM

- ▶ ROM nguyên thủy (Ordinary ROM):
 - ROM các thế hệ đầu tiên
 - Sử dụng tia cực tím để ghi thông tin
- ▶ PROM (Programmable ROM)
 - ROM có thể lập trình
 - Thông tin có thể được ghi vào PROM nhờ thiết bị đặc biệt gọi là bộ lập trình PROM

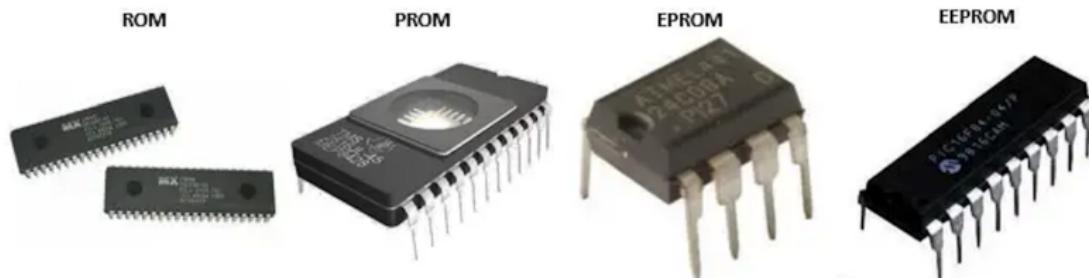


► EPROM (Erasable programmable read-only memory)

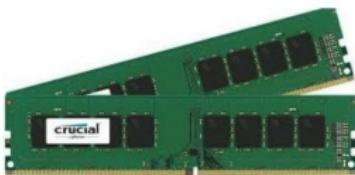
- Là ROM có thể lập trình và xóa được
- Thông tin trong EPROM có thể xóa bằng cách chiếu các tia cực tím có cường độ cao

► EEPROM (Electrically Erasable PROM) :

- Là PROM có thể xóa được thông tin bằng điện
- Có thể ghi được thông tin sử dụng phần mềm chuyên dụng



- ▶ RAM (Random Access Memory) là bộ nhớ truy cập ngẫu nhiên
 - Ô nhớ có thể được truy cập một cách ngẫu nhiên không theo trật tự
 - Tốc độ truy cập các ô nhớ là tương đương
- ▶ Là bộ nhớ không ổn định (dễ bay hơi): mọi thông tin lưu trữ sẽ bị mất khi tắt nguồn
- ▶ Là bộ nhớ bán dẫn. Mỗi ô nhớ là một cổng bán dẫn
- ▶ Sử dụng để lưu trữ thông tin hệ thống và của người dùng
 - Thông tin hệ thống: thông tin phần cứng & hệ điều hành
 - Thông tin người dùng: mã lệnh và dữ liệu các chương trình ứng dụng



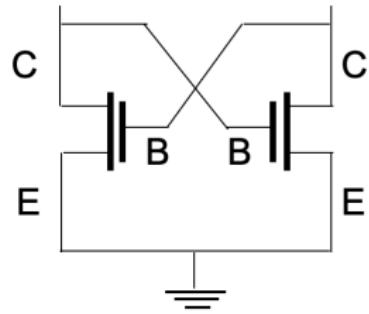
Phân loại bộ nhớ RAM:

► RAM tĩnh (SRAM):

- Mỗi bit của SRAM là một mạch lật flip-flop
- Thông tin lưu trong các bit SRAM luôn ổn định, không cần phải làm tươi định kỳ
- SRAM nhanh nhưng đắt hơn DRAM

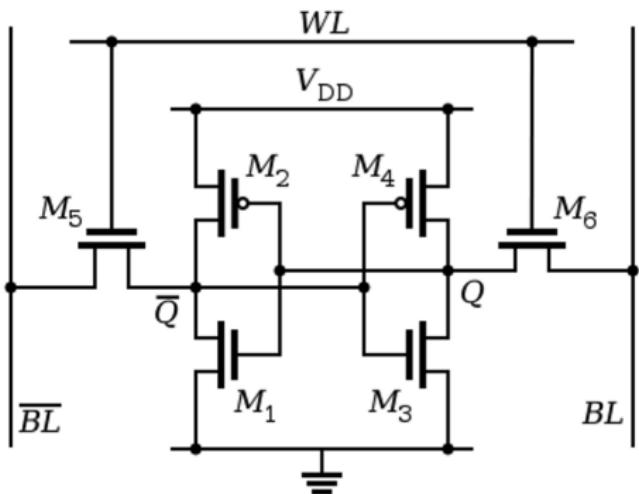
► RAM động (DRAM):

- Mỗi bit của DRAM dựa trên tụ điện
- Thông tin lưu trong bit DRAM không được ổn định, và phải được làm tươi định kỳ
- DRAM chậm hơn SRAM nhưng rẻ hơn



Một mạch lật đơn giản

Cấu tạo của SRAM:

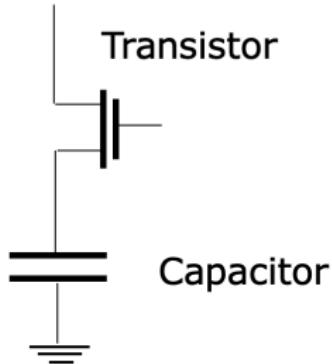


Một ô nhớ SRAM loại 6T

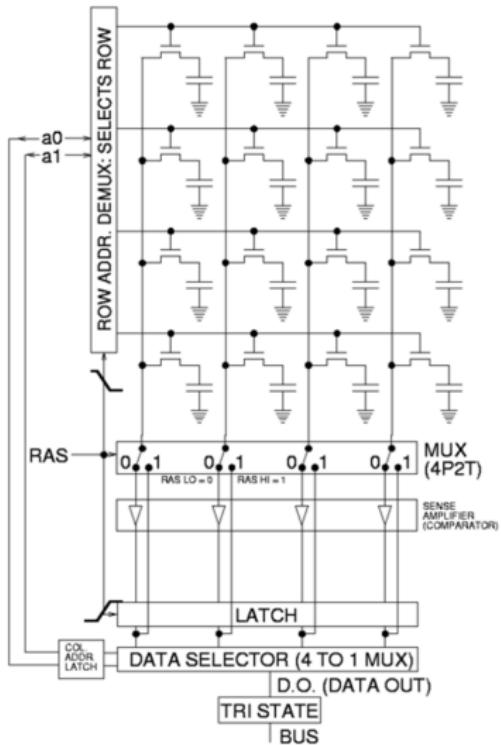
Đặc điểm của SRAM:

- ▶ SRAM sử dụng mạch lật trigơ lưỡng ổn (bistable latching circuit) để lưu 1 bit thông tin
 - Mỗi mạch lật lưu 1 bit thường sử dụng 6, 8, 10 transistor (gọi là mạch 6T, 8T, 10T)
- ▶ SRAM thường có tốc độ truy cập nhanh vì:
 - Các bit có cấu trúc đối xứng
 - Các mạch nhớ SRAM chấp nhận tất cả các chân địa chỉ tại một thời điểm (không dồn kênh)
- ▶ SRAM đắt vì:
 - Nó sử dụng nhiều transistor cho một bit hơn DRAM
 - Vì cấu trúc bên trong phức tạp hơn, mật độ của SRAM thấp hơn DRAM

Cấu tạo của DRAM:



Một bit DRAM



Đặc điểm của DRAM:

- ▶ Bit của DRAM dựa trên 1 tụ điện và 1 transistor.
 - 2 mức tích điện của tụ sẽ biểu diễn 2 mức logic:
 - ▶ Không tích điện: mức 0
 - ▶ Tích đầy điện: mức 1
- ▶ Do tụ thường tự phóng điện, điện tích trong tụ điện có xu hướng bị tổn hao
 - Cần nạp lại thông tin trong tụ thường xuyên để tránh mất thông tin
 - Việc nạp lại thông tin cho tụ là quá trình làm tươi (refresh), phải theo định kỳ

Bộ nhớ Cache

1. Cache là gì?

1.1 Vai trò của cache

1.2 Các nguyên lý cơ bản hoạt động của cache

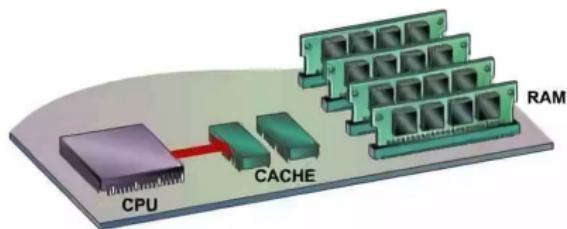
2. Kiến trúc cache

2.1 Tổ chức cache

2.2 Đọc/ ghi trong cache

2.3 Các chính sách thay thế

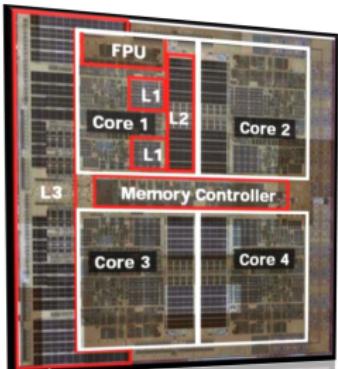
3. Cách thức để nâng cao hiệu năng cache



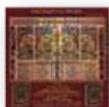
▶ Cache là thành phần nhớ trong sơ đồ phân cấp bộ nhớ máy tính.

- Nó hoạt động như thành phần trung gian, trung chuyển dữ liệu từ bộ nhớ chính về CPU và ngược lại

Một số hình ảnh về bộ nhớ Cache:



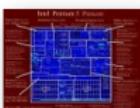
AMD Athlon



■ Intel Quad Core



■ Intel Core i7



■ Intel Pentium 5



■ AMD Quad Core

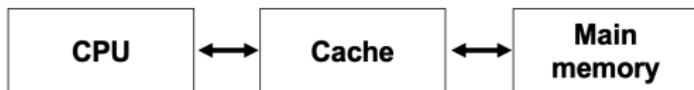
Core L1 SNB: 2.795 MB/s
L1 Cache: 32K 76495 MB/s
L2 Cache: 256K 34720 MB/s
L3 Cache: 3072K 26627 MB/s
Memory: 8176M 16047 MB/s
Processor: Intel(R) Core(TM) i5-2400
Settings: RAM: 0 MHz (DDR3- 0)
752K e820

Bộ nhớ Cache (cont.)

Bộ nhớ Cache là gì?

► Vị trí của cache:

- Với các hệ thống cũ, cache thường nằm ngoài CPU
- Với các CPU mới, cache thường được tích hợp vào trong CPU



► Dung lượng cache thường nhỏ:

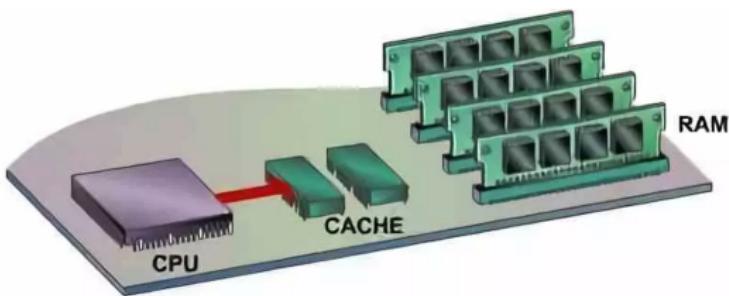
- Với các hệ thống cũ: 16K, 32K,..., 128K
- Với các hệ thống mới: 256K, 512K, 1MB, 2MB, ...

► Tốc độ truy nhập của cache nhanh hơn so với tốc độ bộ nhớ chính

► Giá thành cache (tính trên bit) thường đắt hơn so với bộ nhớ chính

Ví dụ về các thao tác của Cache:

1. CPU yêu cầu nội dung của ngăn nhớ
2. CPU kiểm tra trên cache với dữ liệu này
3. Nếu có, CPU nhận dữ liệu từ cache (nhanh)
4. Nếu không có, đọc Block nhớ chứa dữ liệu từ bộ nhớ chính vào cache
5. Tiếp đó chuyển dữ liệu từ cache vào CPU

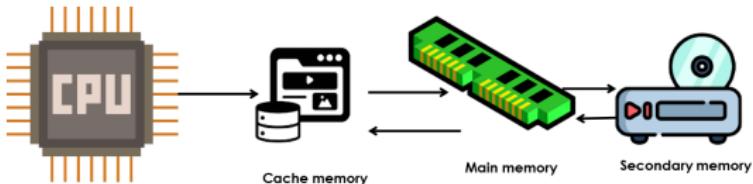


► Nâng cao hiệu năng hệ thống:

- Dung hòa giữa CPU có tốc độ cao và bộ nhớ chính tốc độ thấp (giảm số lượng truy cập trực tiếp của CPU vào bộ nhớ chính)
- Thời gian trung bình CPU truy cập hệ thống bộ nhớ gần bằng thời gian truy cập cache

► Giảm giá thành sản xuất:

- Nếu 2 hệ thống có cùng hiệu năng thì hệ thống có cache sẽ rẻ hơn
- Nếu 2 hệ thống cùng giá thành, hệ thống có cache sẽ nhanh hơn



▶ Cache được coi là **bộ nhớ thông minh**:

- Cache có khả năng đoán trước yêu cầu về lệnh và dữ liệu của CPU
- Dữ liệu và lệnh cần thiết được chuyển trước từ bộ nhớ chính về cache
→ CPU chỉ truy nhập cache → giảm thời gian truy nhập bộ nhớ

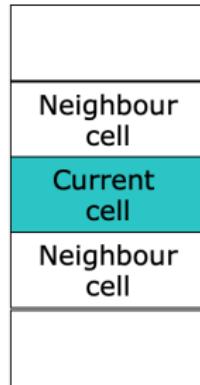
▶ Cache hoạt động dựa trên 2 nguyên lý cơ bản:

- Nguyên lý cục bộ/ **lân cận về không gian** (spatial locality)
- Nguyên lý cục bộ/ **lân cận về thời gian** (temporal locality)



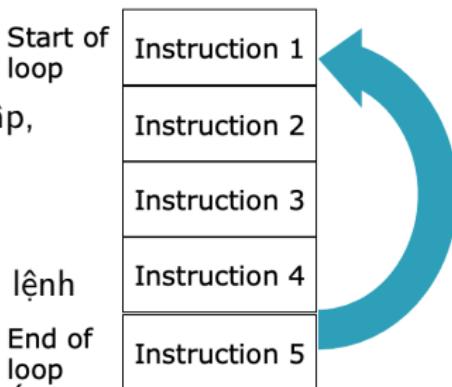
Cục bộ (lân cận) về không gian:

- ▶ Nếu một vị trí bộ nhớ được truy cập, thì xác suất các vị trí gần đó được truy cập trong thời gian gần tới là cao
- ▶ Áp dụng với dữ liệu và các lệnh có thứ tự tuần tự theo chương trình
- ▶ Lệnh trong chương trình thường có thứ tự tuần tự, do đó cache đọc một khối lệnh trong bộ nhớ, mà bao gồm cả các phần tử xung quanh vị trí phần tử hiện tại được truy cập.



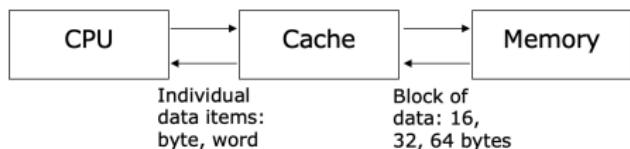
Cục bộ (lân cận) về thời gian:

- ▶ Nếu một vị trí bộ nhớ được truy cập, thì xác suất nó được truy cập lại trong thời gian gần là cao
- ▶ Áp dụng với các mục dữ liệu và các lệnh trong vòng lặp
- ▶ Cache đọc khôi dữ liệu trong bộ nhớ gồm tất cả các thành phần trong vòng lặp



Trao đổi dữ liệu:

- ▶ CPU đọc/ ghi từng mục dữ liệu riêng biệt từ/ vào cache
- ▶ Cache đọc/ ghi khối dữ liệu từ/ vào bộ nhớ

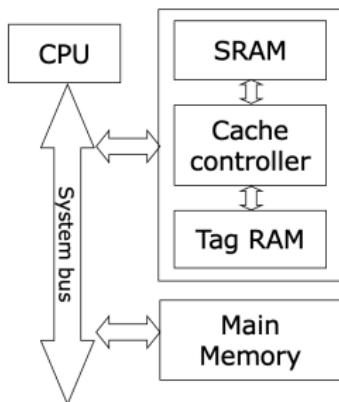


- ▶ **Hit** là sự kiện CPU truy cập dữ liệu/lệnh tìm được trong cache
 - Xác suất xảy ra Hit được gọi là hệ số hit $H: 0 \leq H \leq 1$
 - H càng cao thì cache càng tốt
- ▶ **Miss** là sự kiện CPU truy cập dữ liệu/lệnh không tìm thấy trong cache
 - Khả năng xảy ra Miss gọi là hệ số miss hay $1-H: 0 \leq (1 - H) \leq 1$
 - Hệ số miss thấp thì hiệu quả cache cao

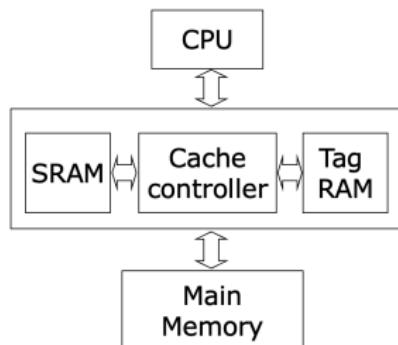
Kiến trúc cache đề cập tới việc Cache được bố trí vào vị trí nào trong mối quan hệ với CPU và Memory.

Hai kiến trúc bố trí của Cache:

- ▶ **Kiến trúc look aside:** Kiến trúc kiểu ngang khàng - bình đẳng
- ▶ **Kiến trúc look through:** Kiến trúc kiểu phân chia - phân cấp



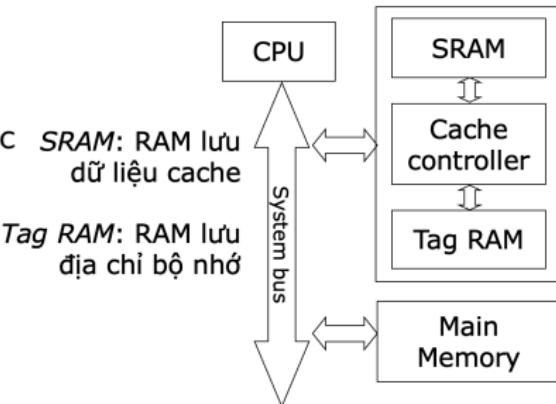
Look aside



Look through

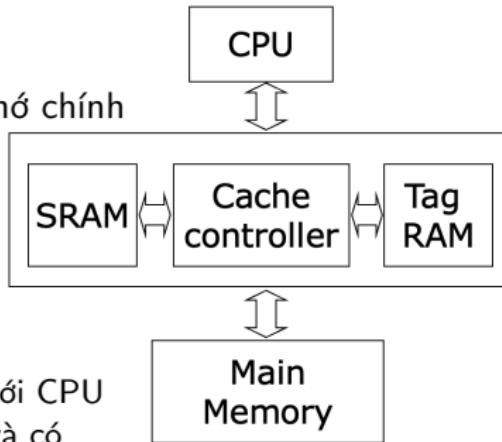
Kiến trúc look aside:

- ▶ Cache và bộ nhớ cùng được **SRAM**: RAM lưu dữ liệu cache kết nối tới bus hệ thống
- ▶ Cache và bộ nhớ chính “thây” chu kỳ bus CPU tại cùng một thời điểm
- ▶ **Ưu điểm:**
 - Thiết kế đơn giản
 - Miss nhanh: khi CPU không tìm thấy mục dữ liệu trong cache nó đồng thời tìm trong bộ nhớ chính cùng 1 chu kỳ xung nhịp
- ▶ **Nhược điểm:**
 - Hit chậm: do cache kết nối với CPU sử dụng bus hệ thống có tần số làm việc không cao và băng thông hẹp



Kiến trúc look through:

- ▶ Cache nằm giữa CPU và bộ nhớ chính



- ▶ Cache “thấy” chu kỳ bus CPU trước sau đó nó “truyền” lại cho bộ nhớ chính

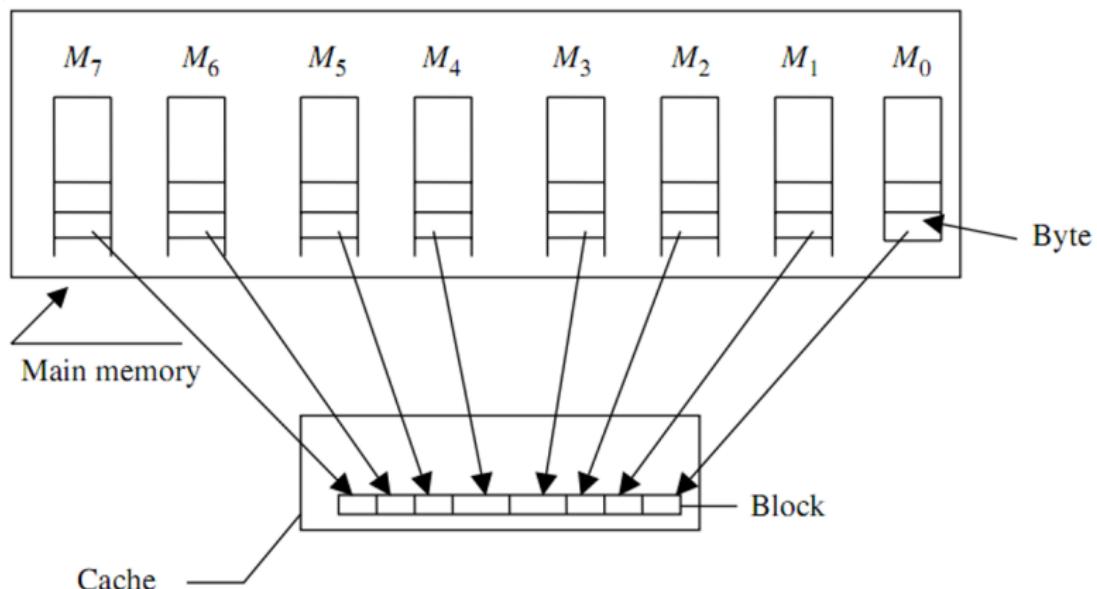
- ▶ Ưu điểm:

- Hit nhanh : Cache kết nối với CPU bằng bus riêng tốc độ cao và có băng thông lớn

- ▶ Nhược điểm:

- Đắt
 - Thiết kế phức tạp
 - Miss chậm : khi CPU không tìm thấy dữ liệu trong cache nó cần tìm trong Bộ nhớ tại 1 xung nhịp tiếp theo

Tổ chức cache giải quyết vấn đề cache và bộ nhớ chính phối hợp làm việc với nhau như thế nào.



Các kỹ thuật tổ chức Cache

► **Ánh xạ trực tiếp** (direct mapping):

- Đơn giản, nhanh
- Ánh xạ cố định

► **Ánh xạ kết hợp đầy đủ** (fully associative mapping):

- Phức tạp, chậm
- Ánh xạ linh hoạt

► **Ánh xạ tập kết hợp/ theo bộ** (set): (set associative mapping):

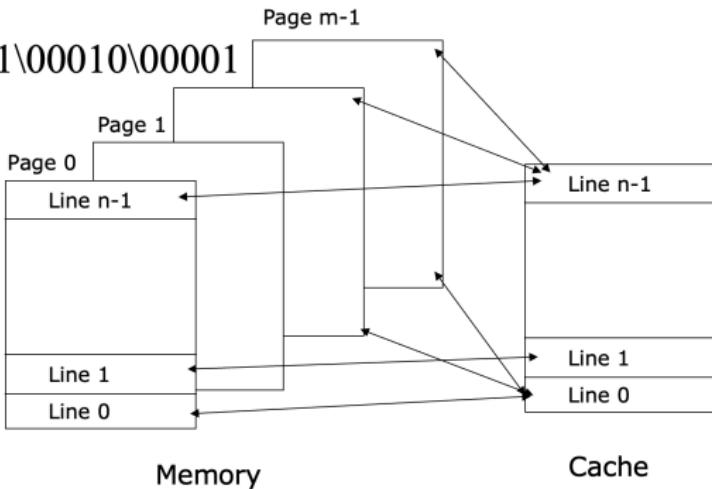
- Phức tạp
- Nhanh, ánh xạ linh hoạt

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Ánh xạ trực tiếp:

0 ... 11\00010\00001



- ▶ **Cache:** Được chia thành n khối hoặc dòng (block or line) từ Line_0 tới Line_{n-1}

▶ Bộ nhớ:

- Được chia thành m trang (page), từ page_0 tới page_{m-1}
 - Mỗi trang bộ nhớ có kích thước bằng cache
 - Mỗi trang có n lines, từ Line_0 tới Line_{n-1}

▶ Ánh xạ:

- Line_i của (page_0 tới page_{m-1}) được ánh xạ tới Line_i của cache

Trường địa chỉ của ánh xạ trực tiếp

Tag	Line	Word
-----	------	------

- ▶ *Tag* (bit): là địa chỉ của page - trang trong bộ nhớ
- ▶ *Line* (bit): là địa chỉ của line - dòng trong một trang cache
- ▶ *Word* (bit): là địa chỉ của word - từ nhớ trong một line

Trong đó:

- ▶ $Tag + Line + Word = Address_Bus_Size \text{ (bit)} = \log_2 \frac{M_{Memory}}{[Cell]}$
- ▶ Số trang nhớ của bộ nhớ: $P = \frac{M_{Memory}}{M_{Cache}} \Rightarrow tag = \log_2 P$
- ▶ Số dòng (line) trong một trang: $L = \frac{M_{Cache}}{[Word]} \Rightarrow line = \log_2 L$
- ▶ Số ô nhớ trong một dòng (line): $W = \frac{[Word]}{[Cell]} \Rightarrow Word = \log_2 W$

Ví dụ: Tìm kích thước trường địa chỉ dùng ánh xạ trực tiếp biết:

- ▶ Kích thước bộ nhớ: 4GB
- ▶ Kích thước cache: 1KB
- ▶ Kích thước của một dòng - line: 32 byte

Giải:

$$M_{Memory} = 4GB = 2^2 * 2^{30}B = 2^{32}B$$

$$\Rightarrow Tag + Line + Word = 32\text{bit}$$

$$M_{Cache} = 1KB = 2^{10}B$$

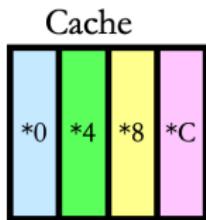
$$[Word] = 32\text{byte} = 2^5B$$

- ▶ Số trang nhớ của bộ nhớ: $P = \frac{M_{Memory}}{M_{Cache}} = \frac{2^{32}B}{2^{10}B} = 2^{22} \Rightarrow tag = 22\text{bit}$
- ▶ Số dòng (line): $L = \frac{M_{Cache}}{[Word]} = \frac{2^{10}B}{2^5B} = 2^5 \Rightarrow line = 5\text{bit}$
- ▶ Số ô nhớ một dòng (line): $W = \frac{[Word]}{[Cell]} = \frac{2^5B}{1B} = 2^5 \Rightarrow Word = 5\text{bit}$

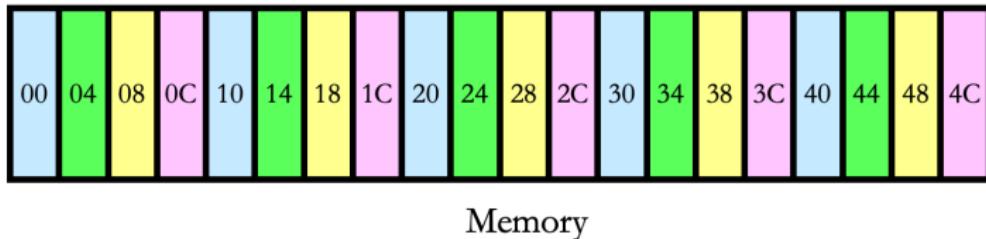
Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ trực tiếp:



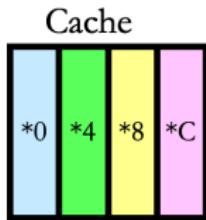
Địa chỉ ánh xạ đến khối / dòng trong bộ nhớ:
 $\text{location} = (\text{Địa chỉ dòng MOD Số dòng trong Cache})$



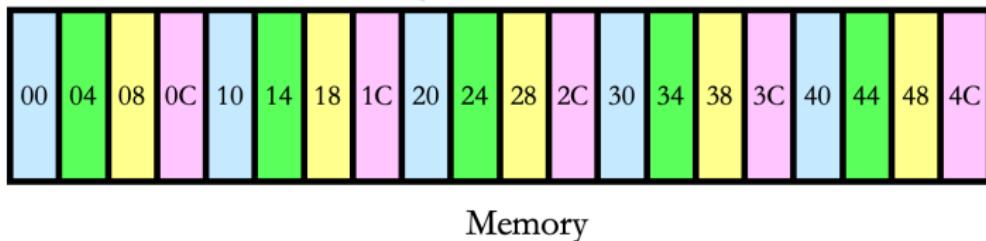
Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ trực tiếp:



Địa chỉ ánh xạ đến khối / dòng trong bộ nhớ:
 $\text{location} = (\text{Địa chỉ dòng MOD Số dòng trong Cache})$

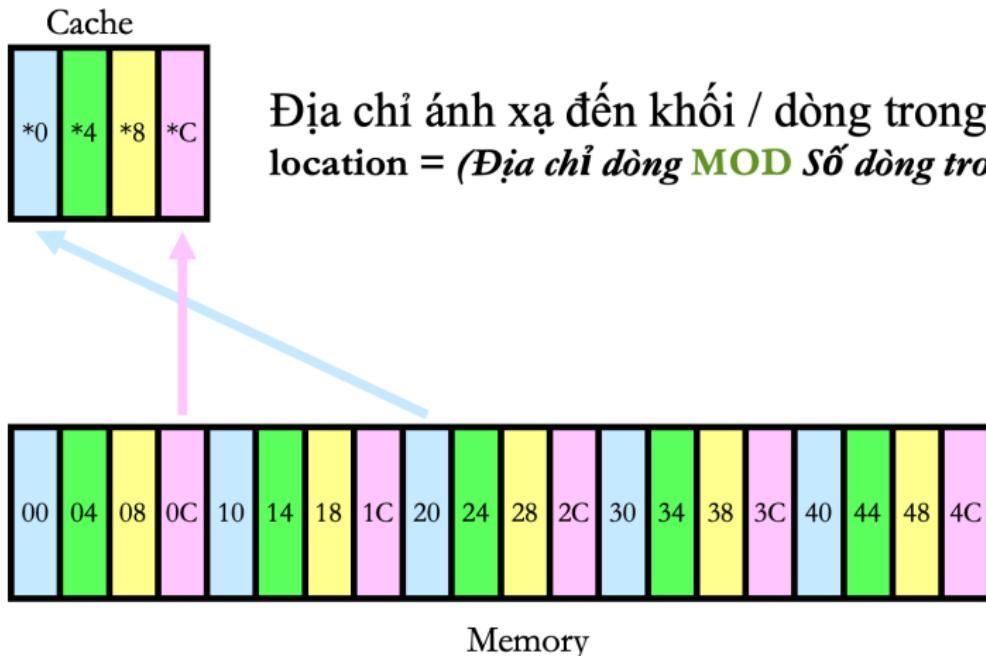


Memory

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ trực tiếp:

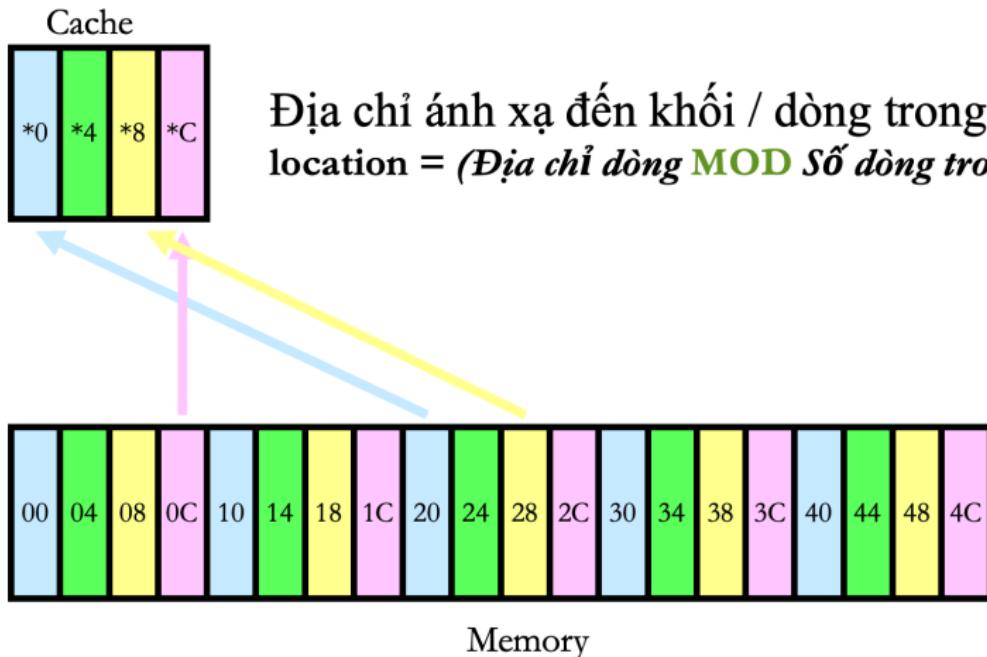


Địa chỉ ánh xạ đến khối / dòng trong bộ nhớ:
 $\text{location} = (\text{Địa chỉ dòng MOD Số dòng trong Cache})$

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ trực tiếp:



Địa chỉ ánh xạ đến khối / dòng trong bộ nhớ:
 $\text{location} = (\text{Địa chỉ dòng MOD Số dòng trong Cache})$

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

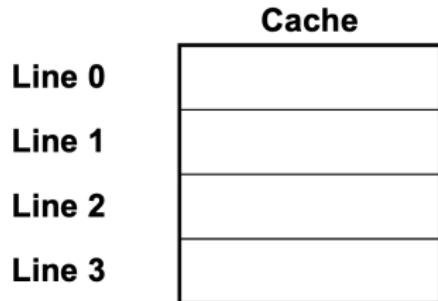


Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 1 - Ánh xạ: khôi 0 $\rightarrow 0 \bmod 4 = 0$

Mem Block	Cache Hit/Miss
0	



Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache



Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 1 - Ánh xạ: khôi 0 $\rightarrow 0 \bmod 4 = 0$

Line 0 trống: write Mem[0]

Mem Block	Cache Hit/Miss
0	Miss

Cache
Mem[0]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 2 - Ánh xạ: khôi 8 $\rightarrow 8 \bmod 4 = 0$

Mem Block	Cache Hit/Miss
0	Miss
8	

Cache
Mem[0]

Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 2 - Ánh xạ: khôi 8 $\rightarrow 8 \bmod 4 = 0$

Line 0 chứa Mem[0]. Ghi đè Mem[8] lên Line 0

Mem Block	Cache Hit/Miss
0	Miss
8	Miss

Cache
Mem[8]

Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 3 - Ánh xạ: khôi 0 $\rightarrow 0 \bmod 4 = 0$

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	

Cache
Mem[8]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache



Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 3 - Ánh xạ: khôi 0 $\rightarrow 0 \bmod 4 = 0$

Line 0 chứa Mem[8]. Ghi đè Mem[0] lên Line 0

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Miss

Cache
Mem[0]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 4 - Ánh xạ: khôi 6 $\rightarrow 6 \bmod 4 = 2$

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Miss
6	

Cache
Mem[0]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache



Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 4 - Ánh xạ: khôi 6 $\rightarrow 6 \bmod 4 = 2$

Line 2 trống: Ghi Mem[6] lên Line 2

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Miss
6	Miss

Cache
Mem[0]
Mem[6]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 5 - Ánh xạ: khôi 8 $\rightarrow 8 \bmod 4 = 2$

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Miss
6	Miss
8	

Cache
Mem[0]
Mem[6]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache



Truy cập bộ nhớ Cache ánh xạ trực tiếp:

Bộ nhớ Cache có Line = 4 và Word = 1 (4 blocks (1-word)). Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 5 - Ánh xạ: khôi 8 → 8 mod 4 = 2

Line 0 chứa Mem[0]: Ghi đè Mem[8] lên Line 0

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Miss
6	Miss
8	Miss

Cache
Mem[8]
Mem[6]

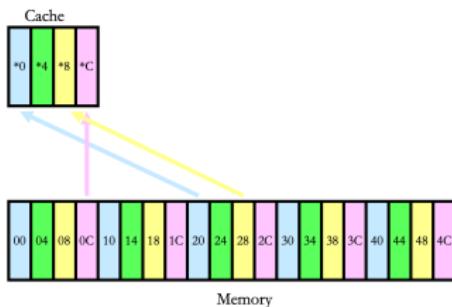
Nhận xét về Cache ánh xạ trực tiếp

► Ưu điểm:

- Thiết kế đơn giản
- Nhanh vì ánh xạ cố định: khi biết địa chỉ bộ nhớ có thể tìm nó trong cache rất nhanh

► Nhược điểm:

- Vì ánh xạ cố định nên khả năng xảy ra xung đột cao
- Tỷ lệ hit thấp

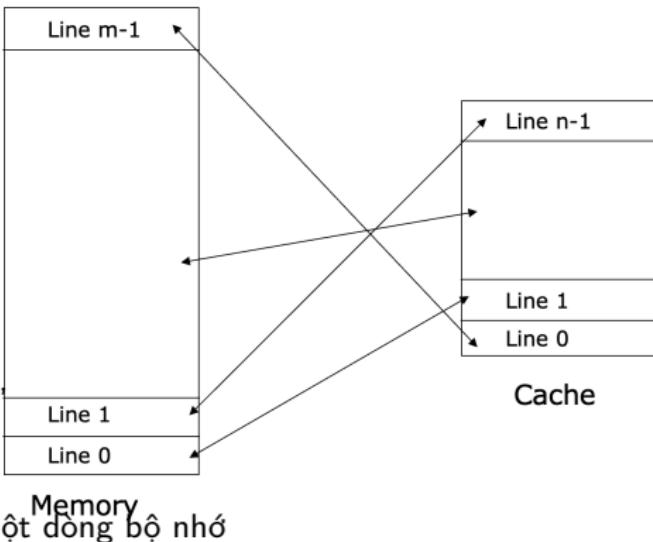


Ánh xạ Kết hợp đầy đủ:

- ▶ **Cache:** Được chia thành n khối hoặc dòng (block or line) từ $Line_0$ tới $Line_{n-1}$

- ▶ **Bộ nhớ:**

- Được chia thành m dòng, từ $Line_0$ tới $Line_{m-1}$
- Kích thước mỗi dòng cache bằng kích thước một dòng bộ nhớ
- Số lượng dòng trong bộ nhớ có thể lớn hơn nhiều số lượng dòng của cache ($m \gg n$)



- ▶ **Ánh xạ:**

- $Line_i$ của bộ nhớ được ánh xạ tới bất kỳ dòng $Line_j$ của cache

Trường địa chỉ của ánh xạ đầy đủ

Tag	Word
-----	------

- ▶ *Tag* (bit): là địa chỉ của line - dòng trong bộ nhớ
- ▶ *Word* (bit): là địa chỉ của word - từ nhớ trong một line

Trong đó:

- ▶ $Tag + Word = Address_Bus_Size \text{ (bit)} = \log_2 \frac{M_{Memory}}{[Cell]}$
- ▶ Số dòng (line) của bộ nhớ: $P = \frac{M_{Memory}}{[Word]} \Rightarrow tag = \log_2 P$
- ▶ Số ô nhớ trong một dòng (line): $W = \frac{[Word]}{[Cell]} \Rightarrow Word = \log_2 W$

Ví dụ: Tìm kích thước trường địa chỉ dùng ánh xạ đầy đủ biết:

- ▶ Kích thước bộ nhớ: 4GB
- ▶ Kích thước cache: 1KB
- ▶ Kích thước của một dòng - line: 32 byte

Giải:

$$M_{Memory} = 4GB = 2^2 * 2^{30}B = 2^{32}B \Rightarrow Tag + Word = 32\text{bit}$$

$$\begin{aligned} M_{Cache} &= 1KB = 2^{10}B \\ [Word] &= 32\text{byte} = 2^5B \end{aligned}$$

- ▶ Số dòng (line) của bộ nhớ: $P = \frac{M_{Memory}}{[Word]} = \frac{2^{32}B}{2^5B} = 2^{27} \Rightarrow tag = 27\text{bit}$
- ▶ Số ô nhớ một dòng (line): $W = \frac{[Word]}{[Cell]} = \frac{2^5B}{1B} = 2^5 \Rightarrow Word = 5\text{bit}$

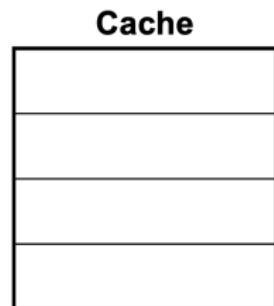
Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache ánh xạ tập đầy đủ:

Quy tắc thay thế dòng cache: Thay thế dòng ít được sử dụng gần đây nhất LRU

Mem Block	Cache Hit/Miss



Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 1:

Line 0 trống: Ghi Mem[0] vào Line 0

Mem Block	Cache Hit/Miss
0	Miss

Cache
Mem [0]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 2:

Mem Block	Cache Hit/Miss
0	Miss
8	

Cache
Mem [0]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 2:

Dòng Line 1-3 trống: Ghi Mem[8] vào Line 1

Mem Block	Cache Hit/Miss
0	Miss
8	Miss

Cache
Mem [0]
Mem[8]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 3:

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	

Cache
Mem[0]
Mem[8]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 3:

Dòng Line 0 chứa Mem[0]

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Hit

Cache
Mem[0]
Mem[8]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 4:

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Hit
6	

Cache
Mem[0]
Mem[8]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 4:

Ghi Mem[6] vào Line 2

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Hit
6	Miss

Cache
Mem[0]
Mem[8]
Mem[6]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 5:

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Hit
6	Miss
8	

Cache
Mem[0]
Mem[8]
Mem[6]

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache



Truy cập bộ nhớ Cache ánh xạ đầy đủ:

Bộ nhớ Cache có Word = 1. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 5:

Line 1 chứa Mem[8]

Mem Block	Cache Hit/Miss
0	Miss
8	Miss
0	Hit
6	Miss
8	Hit

Cache
Mem[0]
Mem[8]
Mem[6]

Nhận xét về Cache ánh xạ đầy đủ

► Ưu điểm:

- Ít xung đột vì ánh xạ linh hoạt
- Tỉ lệ hit cao hơn

► Nhược điểm:

- Chậm vì phải tìm kiếm địa chỉ bộ nhớ trong cache
- Phức tạp vì có thêm n bộ so sánh địa chỉ trong cache

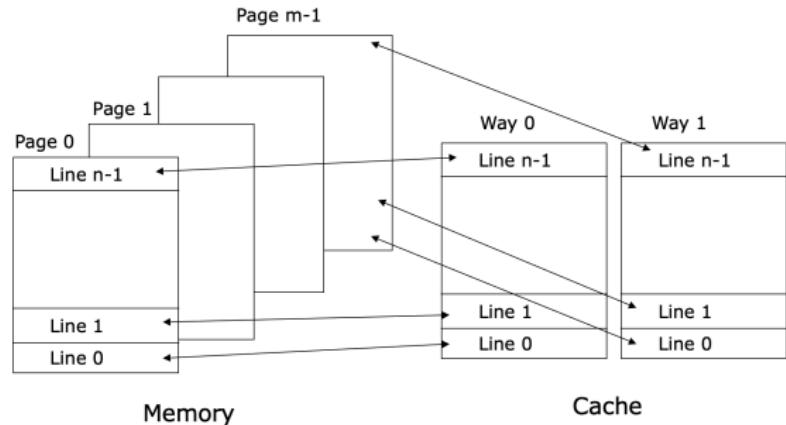
► Thường sử dụng cho cache có kích thước nhỏ

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Ánh xạ tập kết hợp:

- ▶ Được chia thành k đường (ways) có kích thước bằng nhau
- ▶ Mỗi đường được chia thành n dòng (block or line), từ $Line_0$ tới $Line_{n-1}$



Bộ nhớ:

- Được chia thành m trang, từ $page_0$ tới $page_{m-1}$
- Kích thước 1 trang page bằng kích thước 1 way của cache
- Mỗi page được chia thành n dòng (block or line), từ $Line_0$ tới $Line_{n-1}$

► Ánh xạ:

- Page được ánh xạ tới Way (ánh xạ đầy đủ):
 - ▶ Một page bộ nhớ có thể được ánh xạ tới way bất kì của cache
- Line của page ánh xạ tới line của way (ánh xạ trực tiếp - cố định):
 - ▶ $Line_i$ của $page_j$ được ánh xạ tới $Line_i$ của way_j ;

Trường địa chỉ của ánh xạ đầy đủ

Tag	Set	Word
-----	-----	------

- ▶ Tag (bit): là địa chỉ của page - trang trong bộ nhớ
- ▶ Set (bit): là địa chỉ của line - dòng trong một trang nhớ
- ▶ Word (bit): là địa chỉ của word - từ nhớ trong một line

Trường địa chỉ của ánh xạ đầy đủ

Tag	Set	Word
-----	-----	------

Trong đó:

- ▶ $Tag + Set + Word = Address_Bus_Size \text{ (bit)} = \log_2 \frac{M_{Memory}}{[Cell]}$
- ▶ Số trang nhớ của bộ nhớ: $P = \frac{M_{Memory} \times [Way]}{M_{Cache}} \Rightarrow tag = \log_2 P$
- ▶ Số dòng (line) trong một trang: $L = \frac{M_{Cache}}{[Word] \times [Way]} \Rightarrow Set = \log_2 L$
- ▶ Số ô nhớ trong một dòng (line): $W = \frac{[Word]}{[Cell]} \Rightarrow Word = \log_2 W$

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Ví dụ: Tìm kích thước trường địa chỉ dùng ánh xạ tập kết hợp:

- ▶ Kích thước bộ nhớ: 4GB
- ▶ Kích thước cache: 1KB, 2 ways
- ▶ Kích thước của một dòng - line: 32 byte

Giải:

$$M_{Memory} = 4GB = 2^2 * 2^{30}B = 2^{32}B$$

$$\Rightarrow Tag + Line + Word = 32\text{bit}$$

$$M_{Cache} = 1KB = 2^{10}B, [\text{way}] = 2$$

$$[Word] = 32\text{byte} = 2^5B$$

- ▶ Số trang nhớ của bộ nhớ:

$$P = \frac{M_{Memory} \times [\text{Way}]}{M_{Cache}} = \frac{2^{32}B \times 2}{2^{10}B} = 2^{23} \Rightarrow tag = 23\text{bit}$$

- ▶ Số dòng (line): $L = \frac{M_{Cache}}{[Word] \times [\text{Way}]} = \frac{2^{10}B}{2^5B \times 2} = 2^4 \Rightarrow set = 4\text{bit}$

- ▶ Số ô nhớ một dòng (line): $W = \frac{[Word]}{[\text{Cell}]} = \frac{2^5B}{1B} = 2^5 \Rightarrow Word = 5\text{bit}$

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache ánh xạ tập đầy đủ:

Quy tắc thay thế dòng cache: thay thế way ít sử dụng nhất trong cùng một dòng

Mem Block	Cache Hit/Miss
0	

	Way0	Way1
Set 0		
Set 1		

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 1:

Dòng Set 0 trống: Ghi Mem[0] vào Way0

Mem Block	Cache Hit/Miss
0	miss

	Way0	Way1
Set 0	Mem[0]	
Set 1		

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 2:

Mem Block	Cache Hit/Miss
0	miss
8	

	Way0	Way1
Set 0	Mem[0]	
Set 1		

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache



Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 2:

Dòng Set 0 trống: Ghi Mem[8] vào Way1

Mem Block	Cache Hit/Miss
0	miss
8	miss

	Way0	Way1
Set 0	Mem[0]	Mem[8]
Set 1		

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 3:

Mem Block	Cache Hit/Miss
0	miss
8	miss
0	

	Way0	Way1
Set 0	Mem[0]	Mem[8]
Set 1		

Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 3:

Mem Block	Cache Hit/Miss
0	miss
8	miss
0	hit

Dòng Set 0 Way 0 chứa Mem[0]

	Way0	Way1
Set 0	Mem[0]	Mem[8]
Set 1		

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 4:

Mem Block	Cache Hit/Miss
0	miss
8	miss
0	hit
6	

	Way0	Way1
Set 0	Mem[0]	Mem[8]
Set 1		

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 4:

Ghi đè Mem[6] lên dòng Set 0 và Way1

Mem Block	Cache Hit/Miss
0	miss
8	miss
0	hit
6	miss

	Way0	Way1
Set 0	Mem[0]	Mem[6]
Set 1		

Bộ nhớ Cache (cont.)

Tổ chức Bộ nhớ Cache

Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 5:

Mem Block	Cache Hit/Miss
0	miss
8	miss
0	hit
6	miss
8	

	Way0	Way1
Set 0	Mem[0]	Mem[6]
Set 1		

Truy cập bộ nhớ Cache ánh xạ tập kết hợp:

Bộ nhớ Cache có Word = 1, Set = 2, 2 way. Tìm số lượng Miss của bộ nhớ Cache này với chuỗi truy cập khôi/dòng bộ nhớ của CPU là: 0, 8, 0, 6, 8.

- ▶ Truy cập bộ nhớ Cache lần 5:

Ghi đè Mem[8] vào dòng Set0 và Way0

Mem Block	Cache Hit/Miss
0	miss
8	miss
0	hit
6	miss
8	miss

	Way0	Way1
Set 0	Mem[8]	Mem[6]
Set 1		

► Thao tác đọc:

- Trường hợp tìm thấy (hit case): mục dữ liệu yêu cầu tìm thấy trong cache
 - ▶ Mục dữ liệu được đọc từ cache vào CPU
 - ▶ Bộ nhớ chính không tham gia
- Trường hợp không tìm thấy (miss case):
 - ▶ Mục dữ liệu được đọc từ bộ nhớ vào cache
 - ▶ Sau đó dữ liệu được chuyển từ cache vào CPU
- Miss penalty: thời gian truy cập mục dữ liệu bằng tổng thời gian truy cập cache và bộ nhớ chính

► Thao tác ghi:

- Trường hợp hit (bản tin thuộc 1 khối trong cache)
 - ▶ Write through (ghi thẳng): mục dữ liệu được ghi vào cache và ghi vào bộ nhớ đồng thời
 - ▶ Write back (ghi trễ): mục dữ liệu trước tiên được ghi vào cache và cả dòng (block) chứa nó ở trong cache sẽ được ghi lại vào bộ nhớ sau đó, khi mà dòng đó bị thay thế
- Trường hợp miss (bản tin không có trong cache)
 - ▶ Write allocate (ghi có đọc lại): mục dữ liệu trước hết được ghi vào bộ nhớ sau đó cả dòng chứa nó sẽ được đọc vào cache
 - ▶ Write non-allocate (ghi không đọc lại): mục dữ liệu chỉ được ghi vào bộ nhớ

Tại sao phải thay thế dòng cache?

- ▶ Ánh xạ từ 1 dòng trong bộ nhớ → dòng trong bộ nhớ cache thường là ánh xạ nhiều → một
- ▶ Nhiều dòng bộ nhớ chia sẻ một dòng cache → các dòng bộ nhớ được nạp vào cache sử dụng một thời gian và được thay thế bởi dòng khác theo yêu cầu thông tin phục vụ CPU.

Chính sách thay thế (replacement policies): xác định cách thức lựa chọn các dòng trong cache để thay thế khi có dòng mới từ bộ nhớ cần chuyển vào.

Chiến lược thay thế dòng cache gồm 3 chiến lược chính:

- ▶ Thay thế ngẫu nhiên (random replacement)
- ▶ Vào trước ra trước FIFO (First In First Out)
- ▶ Thay thế các dòng ít được sử dụng gần đây nhất LRU (Least Recently Used)

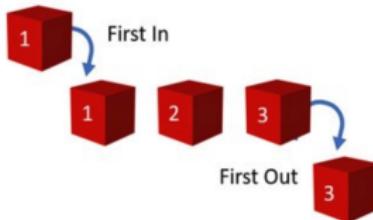
Thay thế ngẫu nhiên (random replacement)

- ▶ Các dòng trong cache được chọn ngẫu nhiên để thay
- ▶ Đơn giản
- ▶ Tỷ lệ miss cao vì phương pháp này không xét tới dòng cache nào đang thực sự được sử dụng
 - Nếu một dòng cache đang được sử dụng mà bị thay thế thì sự kiện miss xảy ra và cần phải đọc lại vào cache



Vào trước ra trước FIFO (First In First Out)

- ▶ Dựa trên nguyên lý FIFO: dòng cache được đọc vào cache trước sẽ được chọn để thay trước
- ▶ Tỷ lệ miss thấp hơn so với phương pháp ngẫu nhiên
- ▶ Tỷ lệ miss vẫn cao vì phương pháp này vẫn chưa thực sự xem xét tới block nào đang thực sự được sử dụng
 - Một dòng cache “cũ” có thể vẫn đang được sử dụng
- ▶ Cài đặt phức tạp vì cần thêm mạch để giám sát thứ tự nạp các dòng bộ nhớ vào cache



Dòng ít được sử dụng gần đây nhất LRU (Least Recently Used)

- ▶ Các dòng cache ít được sử dụng nhất trong thời gian gần đây sẽ được chọn để thay
- ▶ Xét tới các dòng đang được sử dụng
- ▶ Tỷ lệ miss thấp nhất so với phương pháp ngẫu nhiên và FIFO
- ▶ Cài đặt phức tạp vì cần thêm mạch để giám sát tần suất sử dụng các dòng cache



Hiệu năng cache

Thời gian truy cập trung bình của hệ thống nhớ có cache:

$$\begin{aligned} T_{\text{access}} &= (\text{hit cost}) + (\text{miss cost}) \\ &= (\text{Hit cost}) + (\text{miss rate}) * (\text{miss penalty}) \\ &= t_{\text{cache}} + (1 - H) * (t_{\text{memory}}) \end{aligned}$$

Trong đó H là hệ số hit

Nếu $t_{\text{cache}} = 5\text{ns}$, $t_{\text{memory}} = 60\text{ns}$ và $H=80\%$, ta có:

$$t_{\text{access}} = 5 + (1 - 0.8) * (60) = 5+12 = 17\text{ns}$$

Nếu $t_{\text{cache}} = 5\text{ns}$, $t_{\text{memory}} = 60\text{ns}$ và $H=95\%$, ta có:

$$t_{\text{access}} = 5 + (1 - 0.95) * (60) = 5+3 = 8\text{ns}$$

Các yếu tố ảnh hưởng đến hiệu năng cache bao gồm:

- ▶ Kích thước cache
- ▶ Cache nhiều mức
- ▶ Sự phân chia cache

Trong đó:

Yếu tố về kích thước Cache:

- ▶ Kích thước cache lớn:
 - Có thể lưu trữ nhiều khối dữ liệu trong bộ nhớ hơn
 - Giảm tần suất trao đổi các khối dữ liệu của chương trình khác nhau giữa bộ nhớ và cache
 - Cache lớn chậm hơn cache nhỏ:
 - ▶ Tìm kiếm vị trí bộ nhớ trong không gian lớn hơn
 - Xu hướng: cache càng ngày càng lớn
 - ▶ Hỗ trợ đa nhiệm (multi-tasking) tốt hơn

- ▶ Hỗ trợ tốt hơn cho xử lý song song
- ▶ Hỗ trợ tốt hơn cho các hệ thống CPU đa nhân

► Yêu tố Cache nhiều mức:

- Cache nhiều mức có thể dung hòa tốc độ của CPU và MEM tốt hơn cache một mức.

	CPU	L1	L2	L3	Bộ nhớ chính
Cache 3 mức:	1ns	5ns	15ns	30ns	60ns
Cache 1 mức:	1ns	5ns			60ns

- Thực tế, cache thường có 2 mức: L1 và L2. Một số cache có 3 mức: L1, L2 và L3
- Giảm giá thành hệ thống nhớ

► Yêu tố phân chia cache:

- Cache có thể được chia thành cache lệnh và cache dữ liệu để hiệu năng tốt hơn vì:

Bộ nhớ Cache (cont.)

Cách thức để nâng cao hiệu năng của Cache



- ▶ Dữ liệu và lệnh khác nhau về tính cục bộ :
*Dữ liệu có tính chất cục bộ về thời gian hơn
Lệnh có tính chất cục bộ về không gian hơn*
- ▶ Cache lệnh chỉ hỗ trợ thao tác đọc còn cache dữ liệu hỗ trợ cả 2 thao tác đọc/ghi nên *Để tối ưu cache hơn*
- ▶ Hỗ trợ nhiều thao tác đọc/ghi cùng lúc nên *giảm xung đột tài nguyên*
- ▶ Kết hợp các chức năng khác như giải mã trước lệnh vào trong cache lệnh nên *xử lý lệnh tốt hơn*
- Trong thực tế, hầu hết cache L1 được chia thành 2 phần:
 - ▶ I_cache (Instruction cache): cache lệnh
 - ▶ D_cache (Data cache): cache dữ liệu.
- Các cache mức cao hơn không được chia:
 - ▶ Chia cache L1 có lợi ích cao nhất vì nó gần CPU nhất. CPU đọc/ghi trực tiếp lên cache L1. Cache L1 cần hỗ trợ nhiều lệnh truy nhập đồng thời và các biện pháp tối ưu hóa
 - ▶ Chia các cache mức cao hơn không đem lại hiệu quả cao và làm phức tạp trong hệ thống điều khiển cache

Các biện pháp giảm miss:

► Cache tốt:

- Hệ số hit cao
- Hệ số miss thấp
- Miss penalty không quá chậm

► Các kiểu miss:

- *Compulsory misses* (không tìm thấy bắt buộc): thường xảy ra tại thời điểm chương trình được kích hoạt, khi mã chương trình đang được tải vào bộ nhớ và chưa được nạp vào cache
- *Capacity misses* (không tìm thấy do dung lượng): do kích thước của cache hạn chế, đặc biệt trong môi trường đa nhiệm. Khi cache nhỏ, mã chương trình thường xuyên bị chuyển đổi giữa cache và bộ nhớ
- *Conflict misses* (không tìm thấy do xung đột): do có xung đột khi có nhiều dòng nhớ cùng cạnh tranh 1 dòng cache

Các biện pháp giảm miss:

► Tăng kích thước dòng cache:

- Giảm compulsory misses: dòng có kích thước lớn sẽ có khả năng bao phủ lân cận tốt hơn
- Tăng conflict miss: Kích thước dòng lớn => giảm số lượng dòng trong cache => có nhiều dòng bộ nhớ cùng tham chiếu tới dòng cache hơn => tăng conflict miss
- Kích thước dòng lớn có thể gây lãng phí dung lượng của cache
- Kích thước dòng thường dùng là 64 bytes

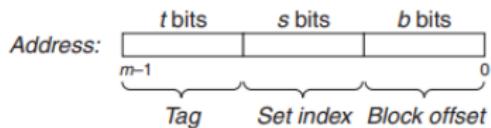
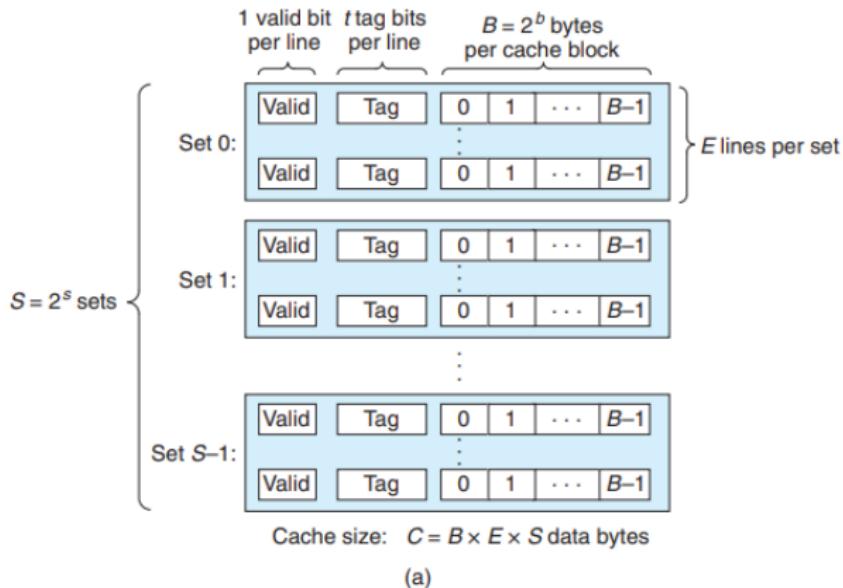
► Tăng mức độ liên kết (tăng số lượng cache way):

- Giảm conflict miss: Tăng số lượng ways => ánh xạ bộ nhớ - cache linh hoạt hơn hay nhiều lựa chọn hơn => giảm conflict miss
- Làm cache chậm hơn: Tăng số lượng ways => không gian tìm kiếm lớn hơn => làm cache chậm đi

Tính toán số bit trong bộ đệm:

- ▶ 1. Tính tổng số bit cần thiết cho một bộ nhớ cache tổ chức theo ánh xạ trực tiếp có kích thước là 64 KByte và một dòng cache có kích thước là 1 từ nhớ, giả sử bus địa chỉ 32 bit?
- ▶ 2. Tổng số bit sẽ cần là bao nhiêu cho bộ nhớ cache sử dụng tập kết hợp được thiết lập 4 way có kích thước là 64 KByte và một dòng cache có kích thước là 1 từ nhớ, giả sử bus địa chỉ 32 bit?
- ▶ 3. Tính tổng số bit cần thiết cho một bộ nhớ cache tổ chức theo ánh xạ trực tiếp kích thước là 64 KB và một dòng cache có kích thước là 8 từ nhớ, giả sử bus địa chỉ 32 bit?

Tính toán số bit trong bộ đệm:



- ▶ 1. Tính tổng số bit cần thiết cho một bộ nhớ cache tổ chức theo ánh xạ trực tiếp kích thước là 64 KByte dữ liệu và một dòng cache có kích thước là 1 từ nhớ, giả sử bus địa chỉ 32 bit?

Giải:

- ▶ Ta có 1 dòng cache 32bit = 4 byte = 2^2 byte $\Rightarrow word = 2$ bit
- ▶ Số dòng Cache là $L = \frac{M_{Cache}}{[Word]} = \frac{64Kbytes}{4byte} = \frac{2^{16}B}{2^2B} = 2^{14} \Rightarrow line = 14bit$
- ▶ $tag = address_size - line - word = 32 - 14 - 2 = 16bit$
- ▶ Số lượng bit cần thiết cho một dòng Cache là:
 $bits/block = data\ bits + tag\ bits + valid\ bit = 32 + 16 + 1 = 49bit$
- ▶ Tổng số bit cần thiết cho bộ nhớ Cache là $= L \times bits/block = 2^{14} \times 49\ bit = 98\ Kbytes$

Tính toán số bit trong bộ đệm:

- ▶ 2. Tổng số bit sẽ cần là bao nhiêu cho bộ nhớ cache sử dụng tập kết hợp được thiết lập 4 way kích thước là 64 KByte dữ liệu và một dòng cache có kích thước là 1 từ nhớ, giả sử bus địa chỉ 32 bit?

Giải:

- ▶ Ta có 1 dòng cache 32bit = 4 byte = 2^2 byte $\Rightarrow word = 2\ bit$
- ▶ Số dòng Cache là
$$L = \frac{M_{Cache}}{[Word]*Way} = \frac{64Kbytes}{4byte*4} = \frac{2^{16}B}{2^4B} = 2^{12} \Rightarrow line = 12bit$$
- ▶ $tag = address_size - line - word = 32 - 12 - 2 = 18bit$
- ▶ Số lượng bit cần thiết cho một dòng Cache là:
 $bits/block = data\ bits + tag\ bits + valid\ bit = 32 + 18 + 1 = 51bit$
- ▶ Tổng số bit cần thiết cho bộ nhớ Cache là $= 4 \times L \times bits/block = 2^{14} \times 51\ bit = 102\ Kbytes$

Tính toán số bit trong bộ đệm:

- ▶ 3. Tính tổng số bit cần thiết cho một bộ nhớ cache tổ chức theo ánh xạ trực tiếp kích thước là 64 KB và một dòng cache có kích thước là 8 từ nhớ, giả sử bus địa chỉ 32 bit?

Giải:

- ▶ Ta có 1 dòng cache $8 \times 32\text{bit} = 32\text{ byte} = 2^5\text{ byte} \Rightarrow word = 5\text{ bit}$
- ▶ Số dòng Cache là $L = \frac{M_{Cache}}{[Word]} = \frac{64\text{Kbytes}}{32\text{byte}} = \frac{2^{16}B}{2^5B} = 2^{11} \Rightarrow line = 11\text{bit}$
- ▶ $tag = address_size - line - word = 32 - 11 - 5 = 16\text{bit}$
- ▶ Số lượng bit cần thiết cho một dòng Cache là:
 $bits/block = data\ bits + tag\ bits + valid\ bit = 8 \times 32 + 16 + 1 = 273\text{bit}$
- ▶ Tổng số bit cần thiết cho bộ nhớ Cache là $= L \times bits/block = 2^{11} \times 273\text{bit} = 68.25\text{ Kbytes}$

Chương 3

- ▶ Bộ nhớ trong và mô hình phân cấp bộ nhớ
- ▶ Phân loại bộ nhớ và tổ chức mạch nhớ
- ▶ ROM
- ▶ RAM
- ▶ Bộ nhớ Cache

Tiếp theo Chương 4 - Lập trình hợp ngữ với bộ vi xử lý 8086/8088

- ▶ Giới thiệu về hợp ngữ
- ▶ Cú pháp của chương trình hợp ngữ
- ▶ Dữ liệu cho chương trình hợp ngữ
- ▶ Biến và hằng
- ▶ Khung chương trình hợp ngữ
- ▶ Các cấu trúc điều khiển

Câu hỏi và bài tập

- **Bài 1: Đề 3 – D20: Câu 2 (2 điểm):** Cho máy có dung lượng bộ nhớ chính: 64KB, dòng cache 8 Bytes, bộ nhớ cache được gồm 32 dòng (lines) được tổ chức ánh xạ trực tiếp.
1. Xác định số bit các thành phần địa chỉ của ô nhớ.
 2. Ô nhớ có địa chỉ 0D20H trong bộ nhớ chính được nạp vào dòng (lines) nào của cache
- **Bài 2: Câu 3 - Đề 2 - 2018:** Giả sử kiến trúc máy tính 32 bit. Tính kích thước dung lượng của bộ nhớ cache có thể chứa 64Kbyte data, và mỗi khôi trong bộ nhớ chứa 8 word data với phương pháp ánh xạ sau:
1. Ánh xạ trực tiếp
 2. Ánh xạ tập kết hợp 4 đường (4 way set-associative)
 3. So sánh hai phương pháp trên?

Câu hỏi và bài tập (cont.)

► **Bài 3:** Giả thiết rằng máy tính có 128KiB cache tổ chức theo kiểu ánh xạ liên kết tập hợp 4 way. Cache có tất cả là 1024 Set từ S0 đến S1023. Địa chỉ bộ nhớ chính là 32 bit và đánh địa chỉ cho từng byte.

1. Tính số bit cho các trường địa chỉ khi truy nhập cache ?
2. Xác định byte nhớ có địa chỉ D202023A(H) được ánh xạ vào Set nào của cache ?

► **Bài 4:** Máy tính dùng 32 bit địa chỉ để đánh địa chỉ cho bộ nhớ theo byte; bus dữ liệu để kết nối với bộ nhớ chính là 32 bit. Hãy cho biết:

1. Số byte nhớ tối đa được đánh địa chỉ? Địa chỉ đầu và địa chỉ cuối dưới dạng Hexa?
2. Hãy cho biết các byte nhớ có địa chỉ sau đây 0FE12C3D(H), 10ABCD06(H) được bố trí ở dòng nhớ (line) nào?

Câu hỏi và bài tập (cont.)

► **Bài 5:** Cho máy tính với 64Kbytes bộ nhớ chính được đánh địa chỉ theo byte, bộ nhớ cache gồm 32 lines được tổ chức ánh xạ trực tiếp, kích thước mỗi line là 8 bytes.

1. Xác định số bit của các trường địa chỉ: Tag, Line, Word
2. Chỉ ra mỗi byte nhớ của bộ nhớ chính có địa chỉ cho dưới đây được nạp vào line nào của cache:

0001 0001 0001 1011
1100 0011 0011 0100
1101 0000 1101 1101
1010 1010 1010 1010

3. Giả thiết byte nhớ có địa chỉ 0001 1010 0001 1010 được nạp vào cache, hãy chỉ ra địa chỉ theo dạng nhị phân của những byte nhớ khác cùng được nạp với byte nhớ đó trong cùng line.