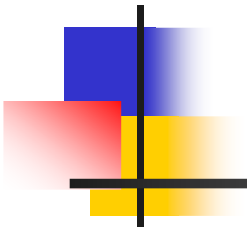
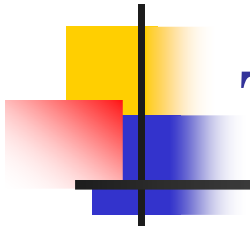
BÀI 9

BỘ NHỚ TRONG

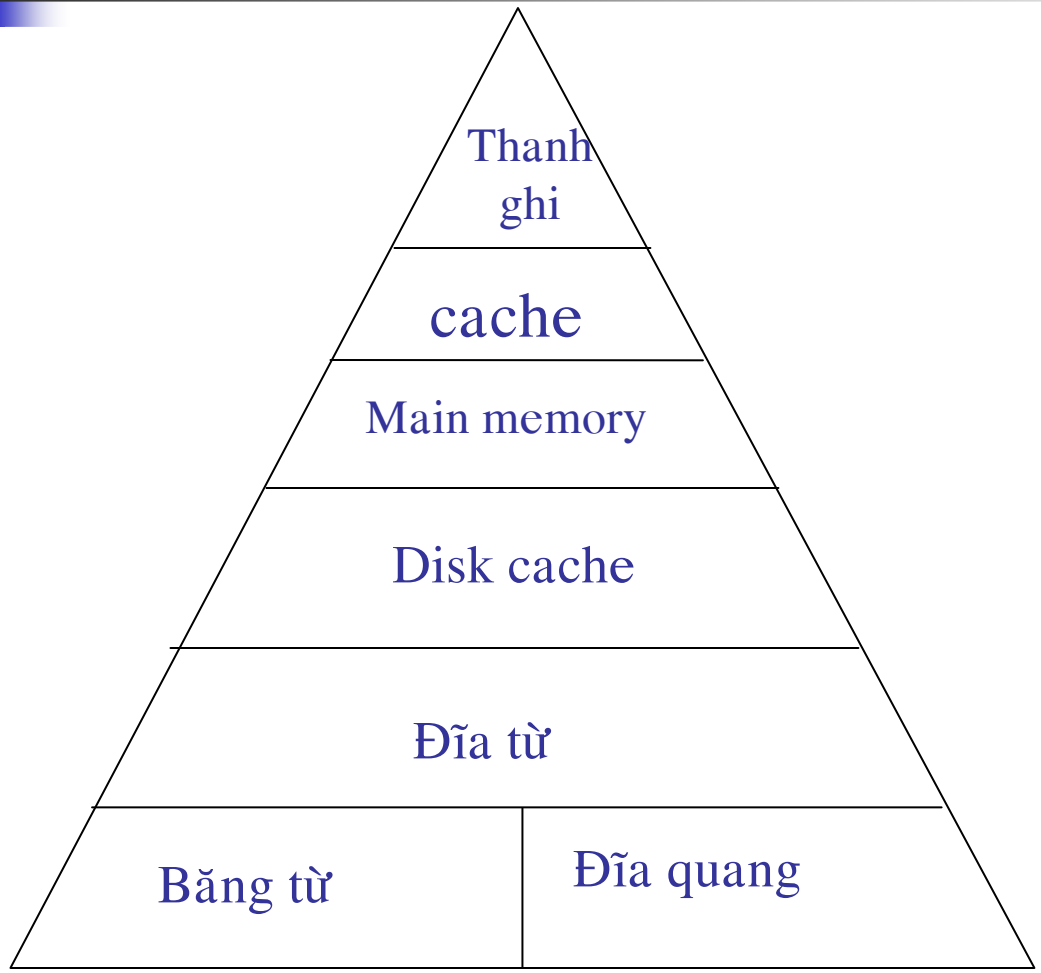




Tổng quan hệ thống bộ nhớ máy tính

- Đặc trưng của hệ thống bộ nhớ
  - Vị trí: trong hay ngoài, trong CPU
  - Dung lượng: kích thước từ nhớ, số lượng từ nhớ
  - Đơn vị truyền tải: từ hay khối
  - Phương pháp truy xuất: truy xuất tuần tự, truy xuất trực tiếp, truy xuất ngẫu nhiên, truy xuất liên kết (cache)
  - Hiệu suất: thời gian truy xuất, tốc độ truyền, chu kỳ
  - Dạng vật lý: bán dẫn hay băng từ
  - Đặc tính vật lý: thay đổi/không thay đổi, xóa được/không thể xóa
  - Tổ chức bộ nhớ: sắp xếp vật lý các bit để hình thành một từ

Phân cấp bộ nhớ



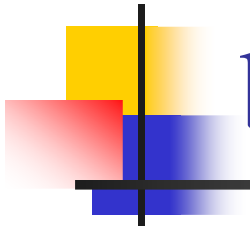
-Giảm giá thành

-Dung lượng tăng

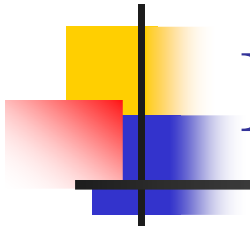
-Thời gian truy xuất tăng

-Tần suất truy xuất của CPU giảm

Bộ nhớ chính_các loại bộ nhớ bán dẫn

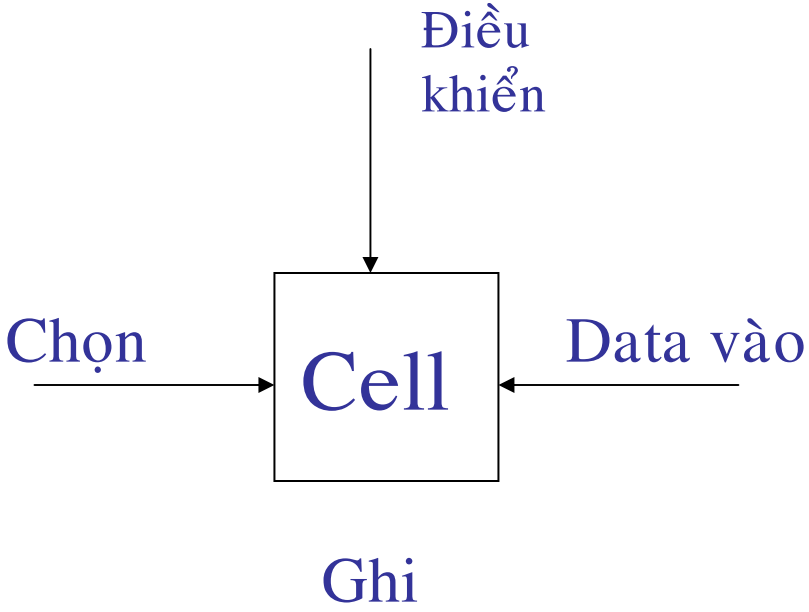


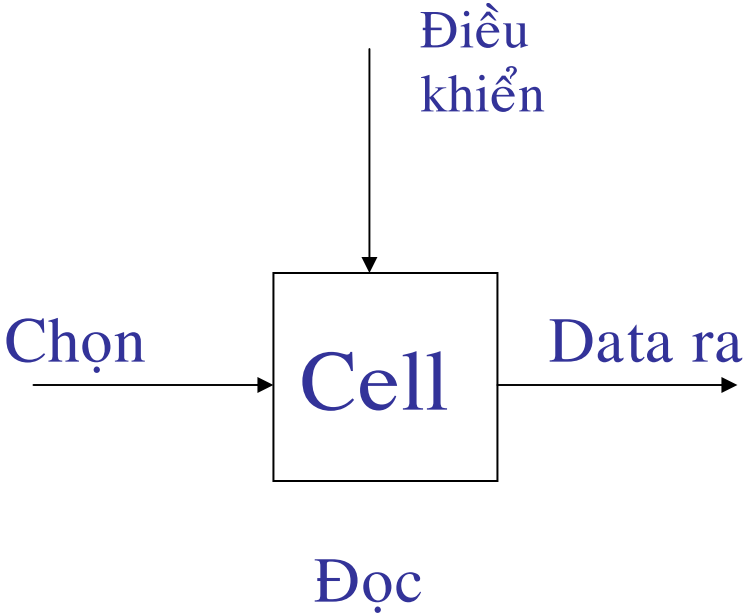
- RAM (Random Access Memory)
  - RAM động: làm từ tụ điện, cần làm tươi, mật độ cao
  - RAM tĩnh: làm bằng các flip-flop, nhanh
- ROM (Read Only Memory)
  - PROM (Programmable ROM)
  - EPROM(Erasable PROM)
  - EEPROM (Electrically EPROM)
  - Flash Memory: lập trình lại rất nhanh, mật độ cao, xóa bằng điện và chỉ cần vài giây.

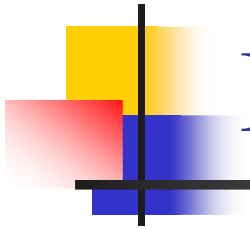


Bộ nhớ chính_tổ chức bộ nhớ

- Cell nhớ là phần tử cơ bản có các thuộc tính:
  - Hai trạng thái: 1 và 0
  - Có thể cài đặt trạng thái, hoạt động ghi
  - Có thể đọc trạng thái, hoạt động đọc

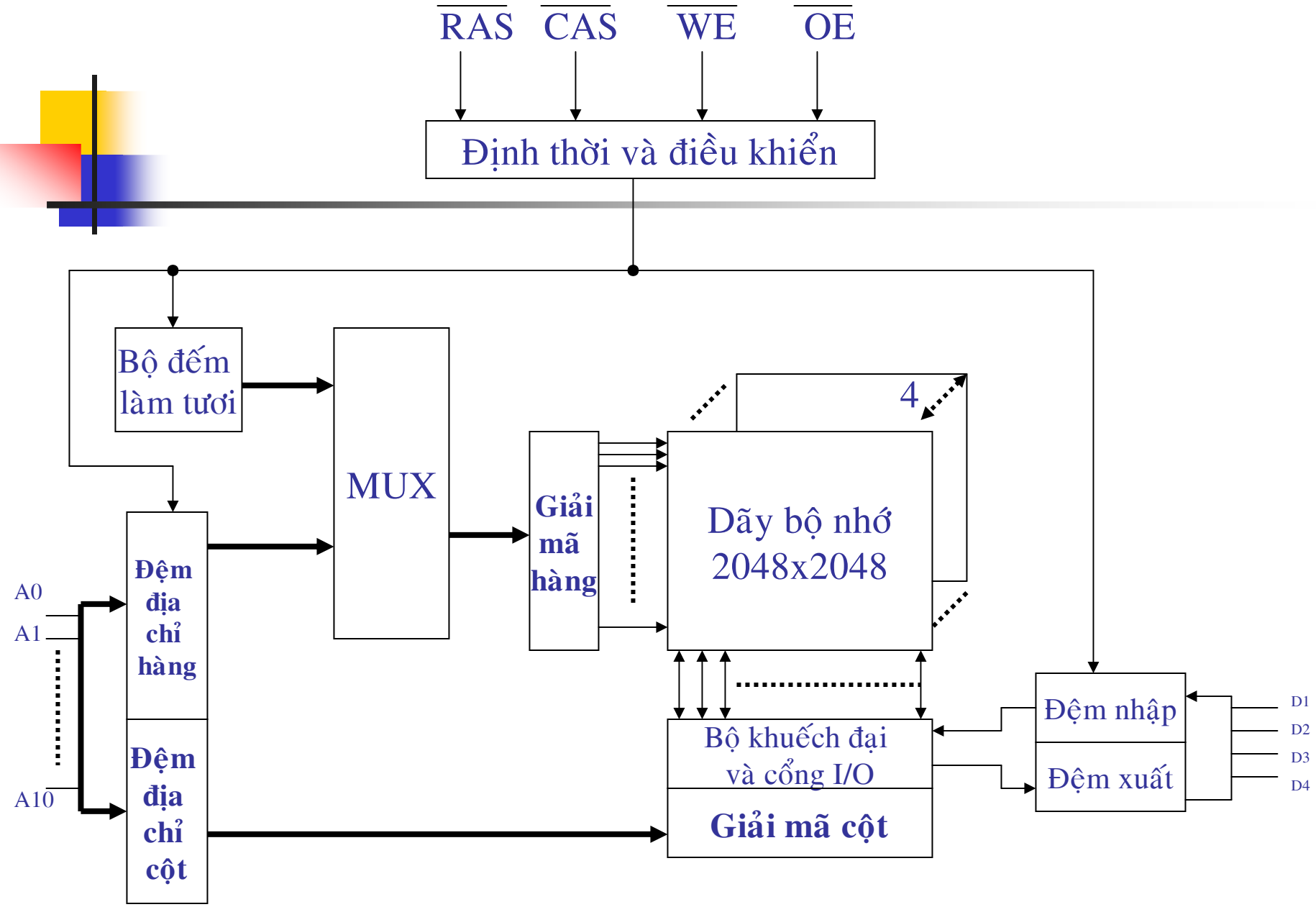

Điều
khiển
Chọn
Cell
Data vào
Ghi

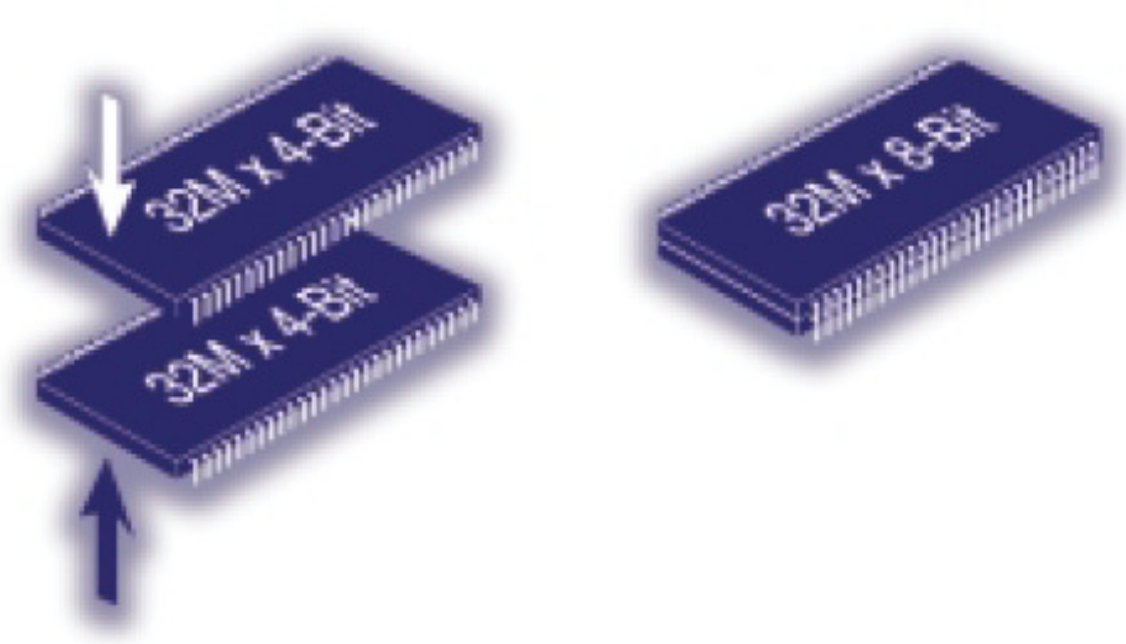

Điều
khiển
Chọn
Cell
Data ra
Đọc

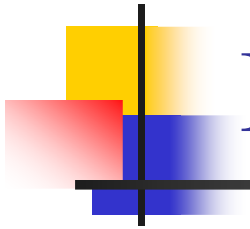


Bộ nhớ chính_chip logic

- Mỗi chip gồm một dãy các cell nhớ
- Dãy được tổ chức thành W từ B bit (vd: 16_Mbit chip được tổ chức từ 1M từ 16 bit)
- 1bit chip: data được đọc ghi mỗi lần một bit

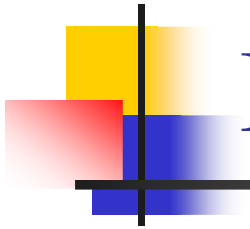

RAS
CAS
WE
OE
Định thời và điều khiển
Bộ đếm làm tươi
MUX
4
Giải mã hàng
Dãy bộ nhớ 2048x2048
A0
A1
A10
Đệm địa chỉ hàng
Đệm địa chỉ cột
Bộ khuếch đại và cổng I/O
Giải mã cột
Đệm nhập
Đệm xuất
D1
D2
D3
D4


32M x 4-Bit
32M x 4-Bit
32M x 8-Bit



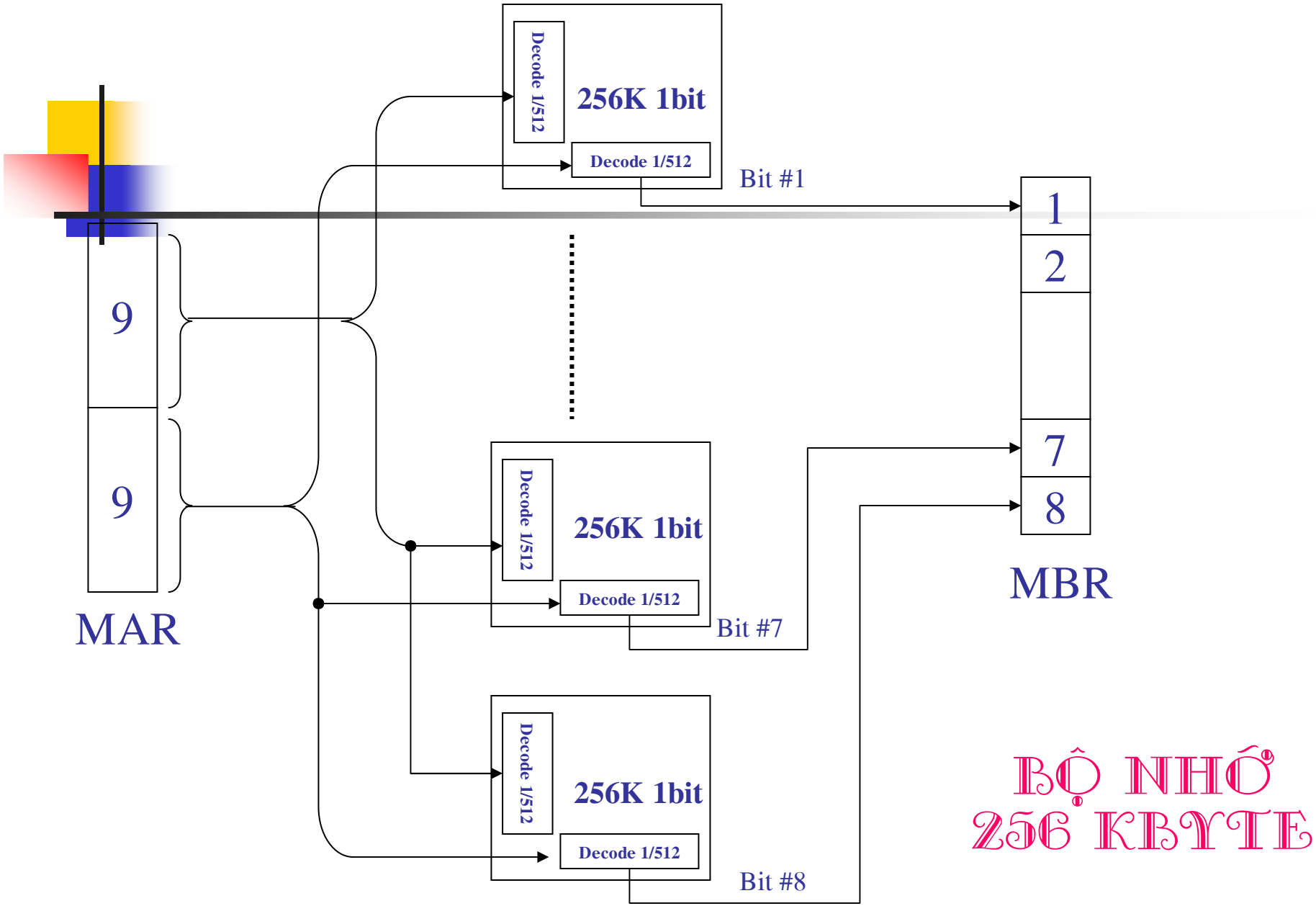
Bộ nhớ chính_chip logic

- RAS (row address select), CAS (column address select)
- Hàng kết nối đến ngõ Chọn (select) của cell
- Cột kết nối đến ngõ vào/ra data
- Số đường địa chỉ cần = $\log_2 W$

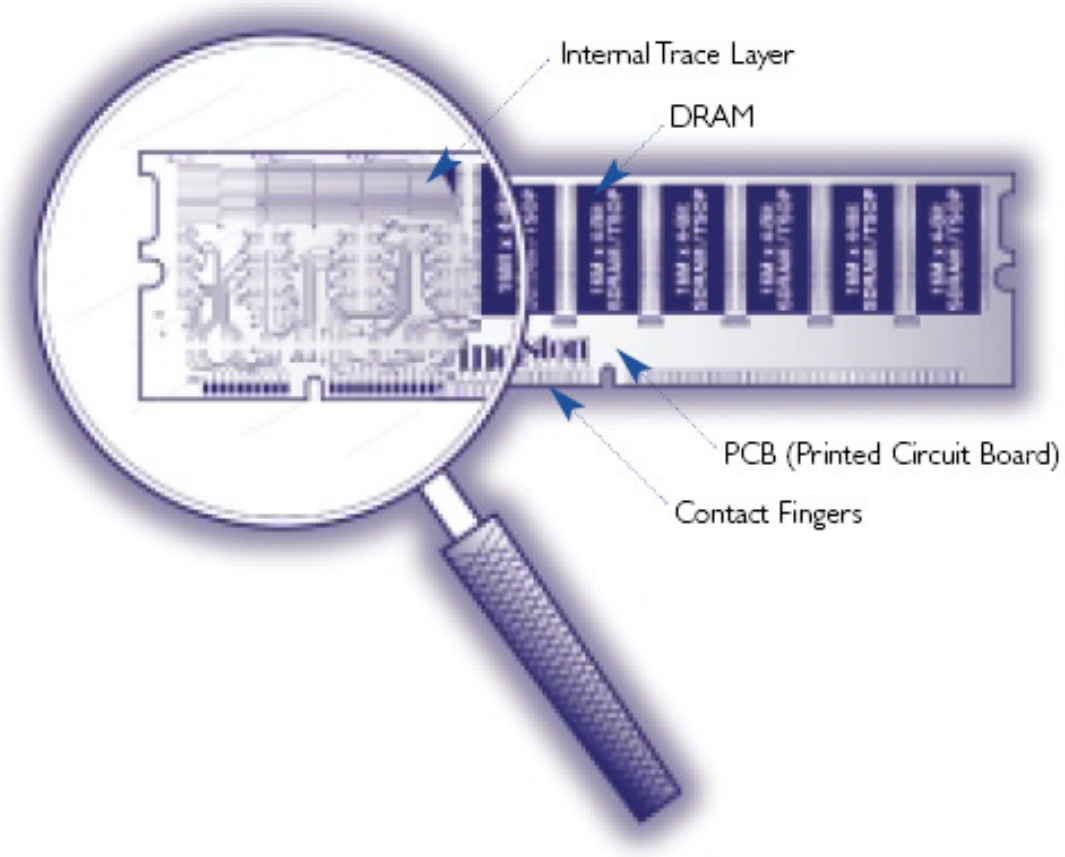


Bộ nhớ chính_tổ chức module

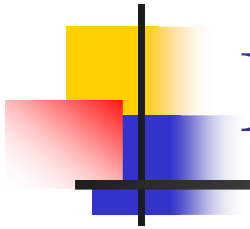
- Nếu chip 1 bit thì cần số chip ít nhất bằng số bit/từ của RAM
- Tổ chức một RAM căn cứ vào dung lượng yêu cầu và loại chip



BỘ NHỚ 256 KBYTE

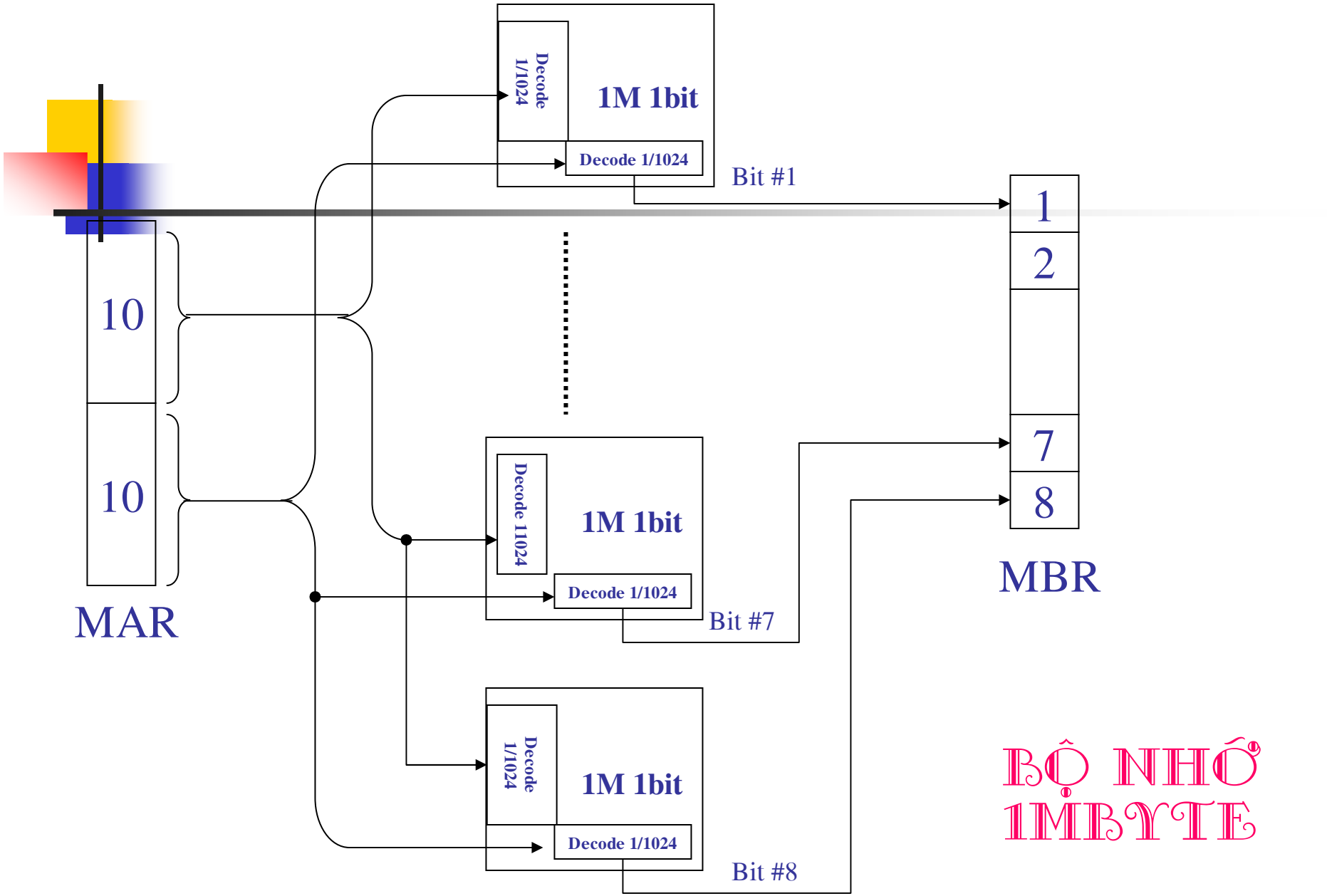


A closer look at a
168-pin SDRAM DIMM.

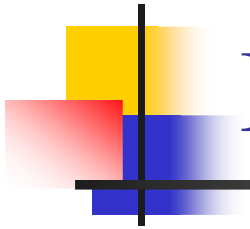


Bài tập

- Thiết kế bộ nhớ DRAM dung lượng 1Mbyte với các chip 1bit (dung lượng một chip là tùy chọn)? Từ nhớ 1 byte.
- Thiết kế DRAM 1Mbyte với các chip 1 bit co dung lượng 256Kbit? Từ nhớ 1 byte.
- Hãy vẽ sơ đồ thiết kế DRAM 256Mbyte với các chip 2 bit có dung lượng 128Mbit? Từ nhớ 1 byte.
- Thiết kế DRAM 1Mbyte với 16 chip 1 bit co dung lượng 256Kbit va cac chip 4 bit co dung luong 1Mbit? Từ nhớ 1 byte.
- Thiết kế bộ nhớ DRAM dung lượng 4Mbyte với các chip 2bit (dung lượng một chip là tùy chọn)



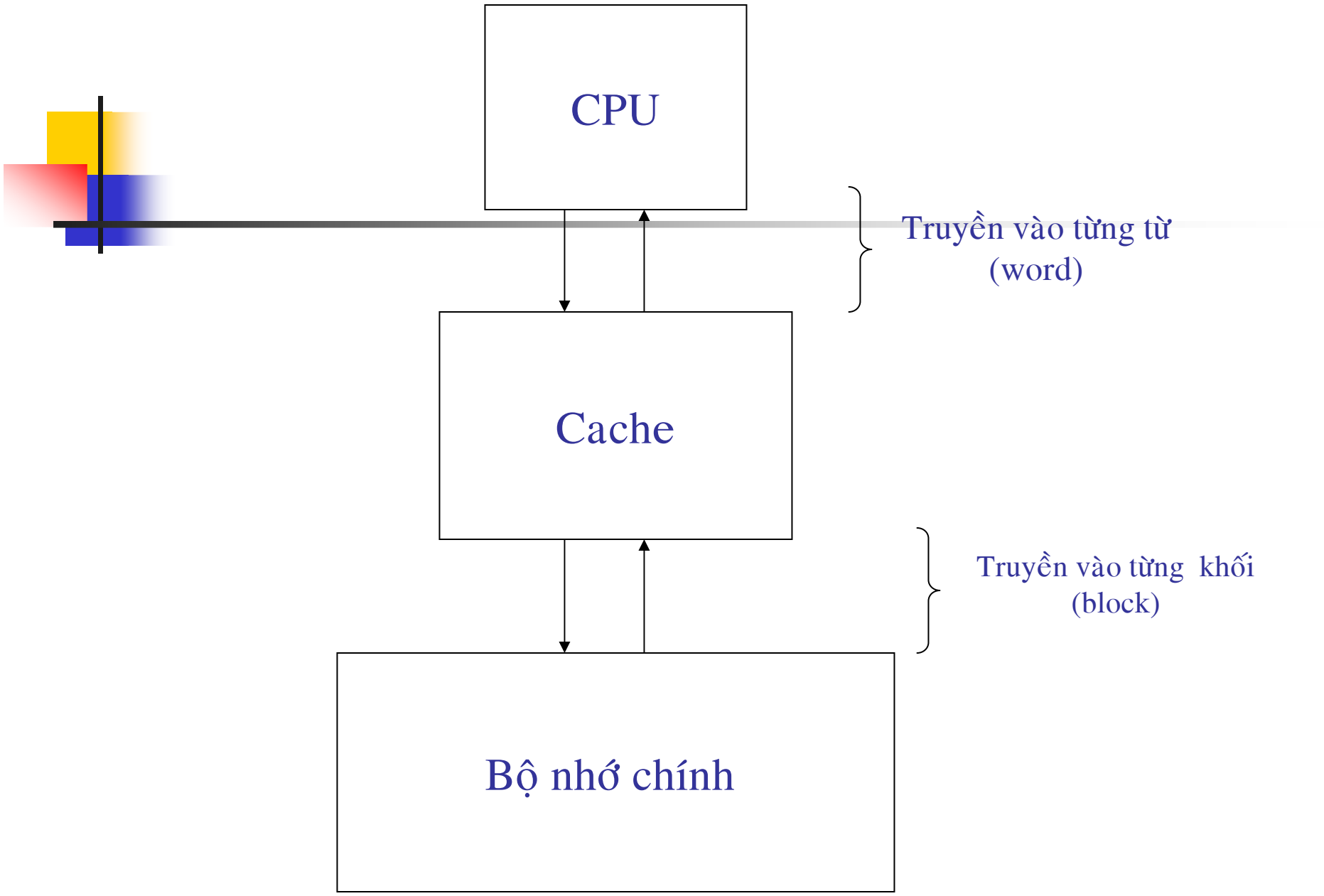
BỘ NHỚ 1MBYTE

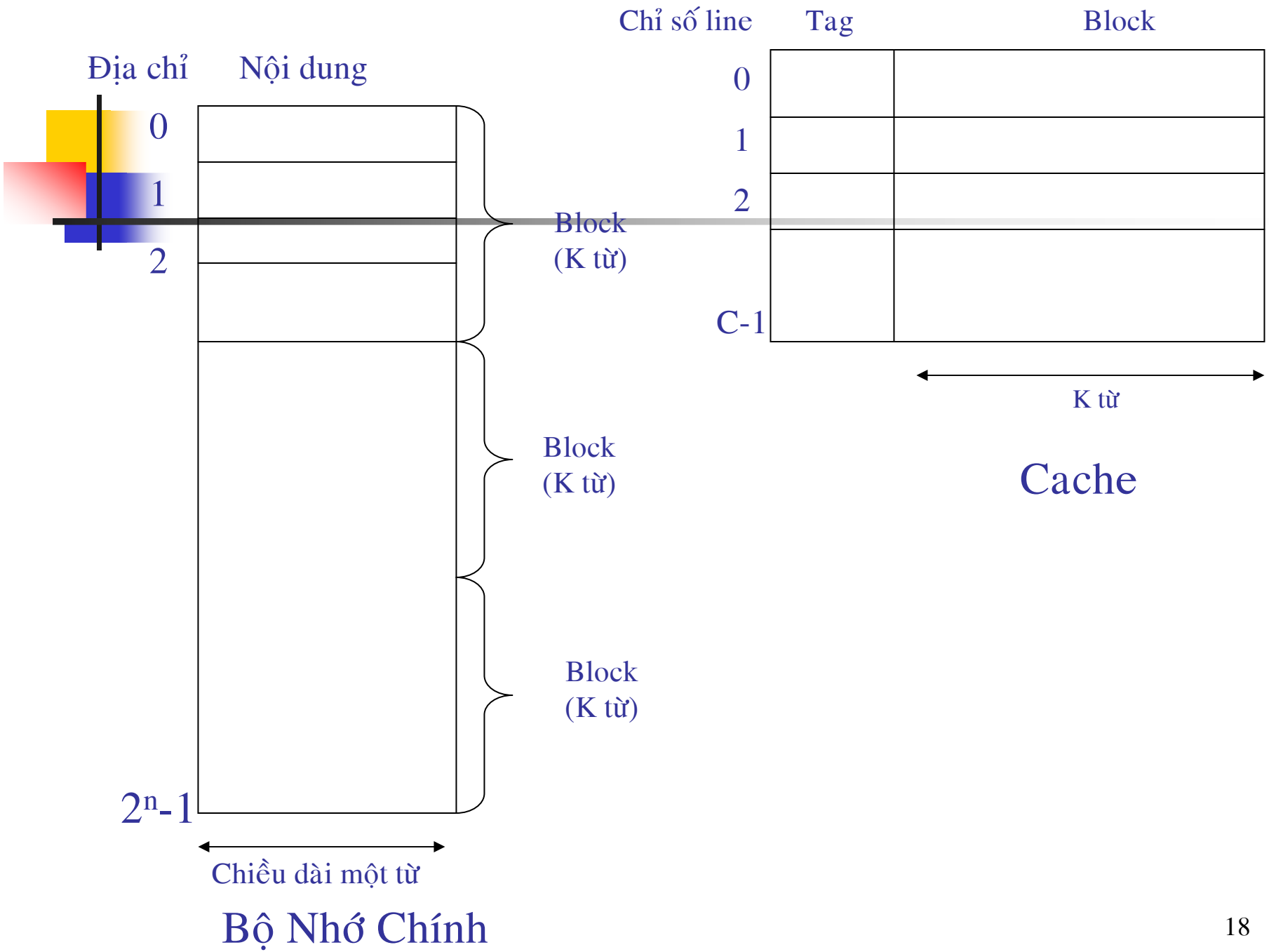


Bộ nhớ cache

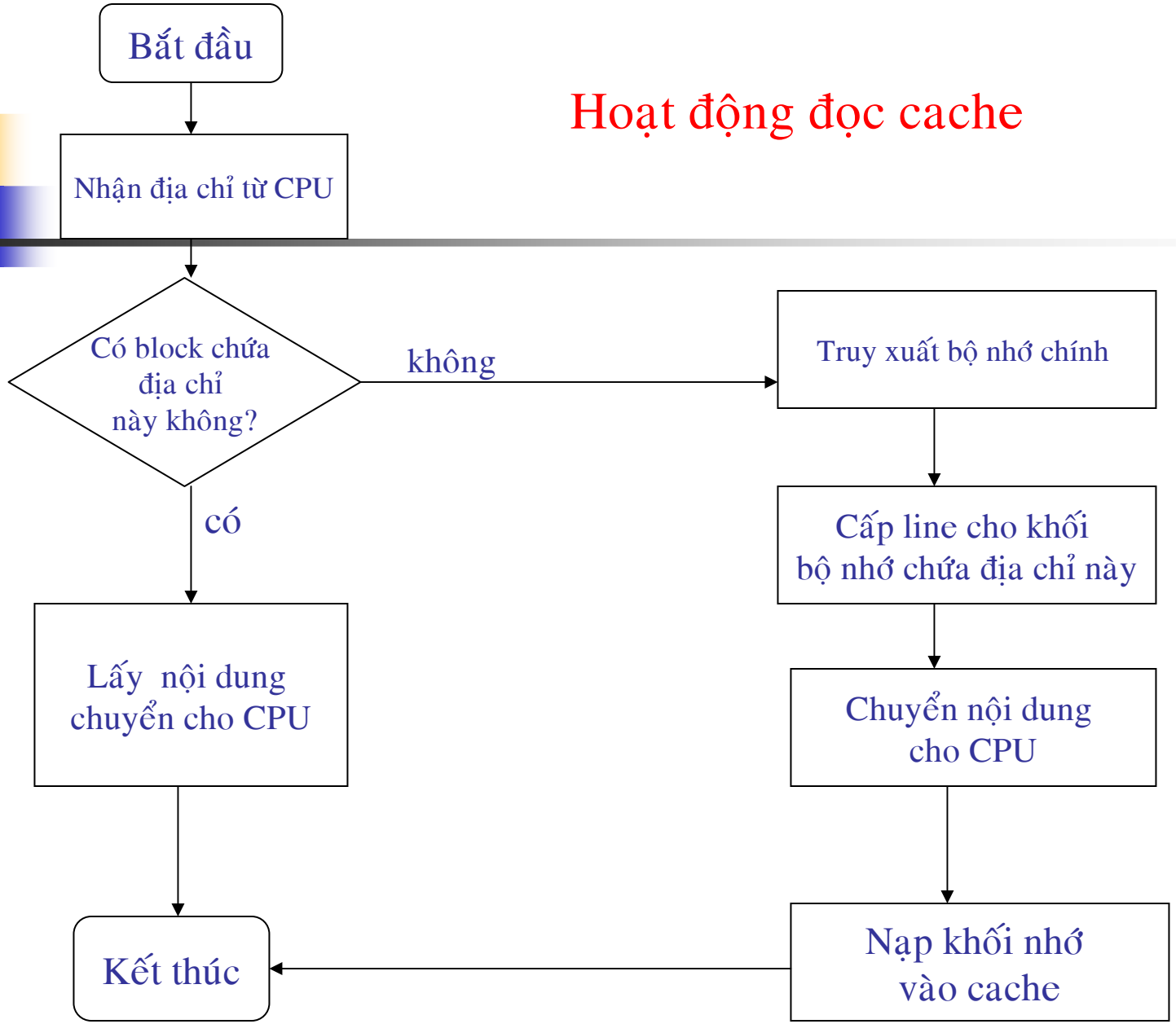
- Nguyên lý:

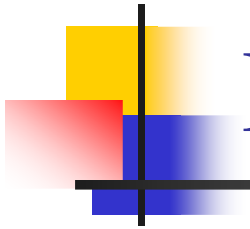
  Bộ nhớ cache được tạo ra để cung cấp giải pháp tăng tốc độ truy xuất bộ nhớ. Cache chứa một phần bộ nhớ chính. Khi CPU muốn đọc một từ trong bộ nhớ chính, trước hết nó kiểm tra xem từ này có trong cache không. Nếu có thì lấy ngay, nếu không thì một khối chứa từ này được nạp vào cache và từ nhớ được cấp cho CPU. Điều này được thực hiện trên cơ sở dự đoán các tham chiếu kế tiếp có từ nhớ thuộc khối này.


CPU
Truyền vào từng từ
(word)
Cache
Truyền vào từng khối
(block)
Bộ nhớ chính


Chỉ số line
Tag
Block
Địa chỉ
Nội dung
0
1
2
C-1
Block
(K từ)
K từ
Cache
Block
(K từ)
Block
(K từ)
2^n-1
Chiều dài một từ
Bộ Nhớ Chính

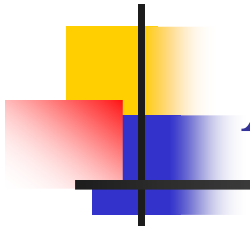
Hoạt động đọc cache





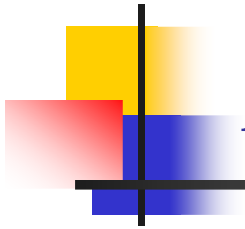
Kích thước của cache

- Bộ nhớ có kích thước 2^n, chia làm M khối, mỗi khối có K từ => $M=2^n/K$
- Cache có C line, mỗi line chứa một tag và một khối K từ
- M>>C
- Cache đủ nhỏ để chi phí ~ main memory và không trở nên chậm
- Cache đủ lớn để truy xuất nhanh
- Phẩm chất cache∈ công nghệ vi mạch
- Hệ số tìm thấy (hit ratio): khả năng lấy được số liệu cần thiết từ cache.



Ánh xạ bộ nhớ chính vào cache

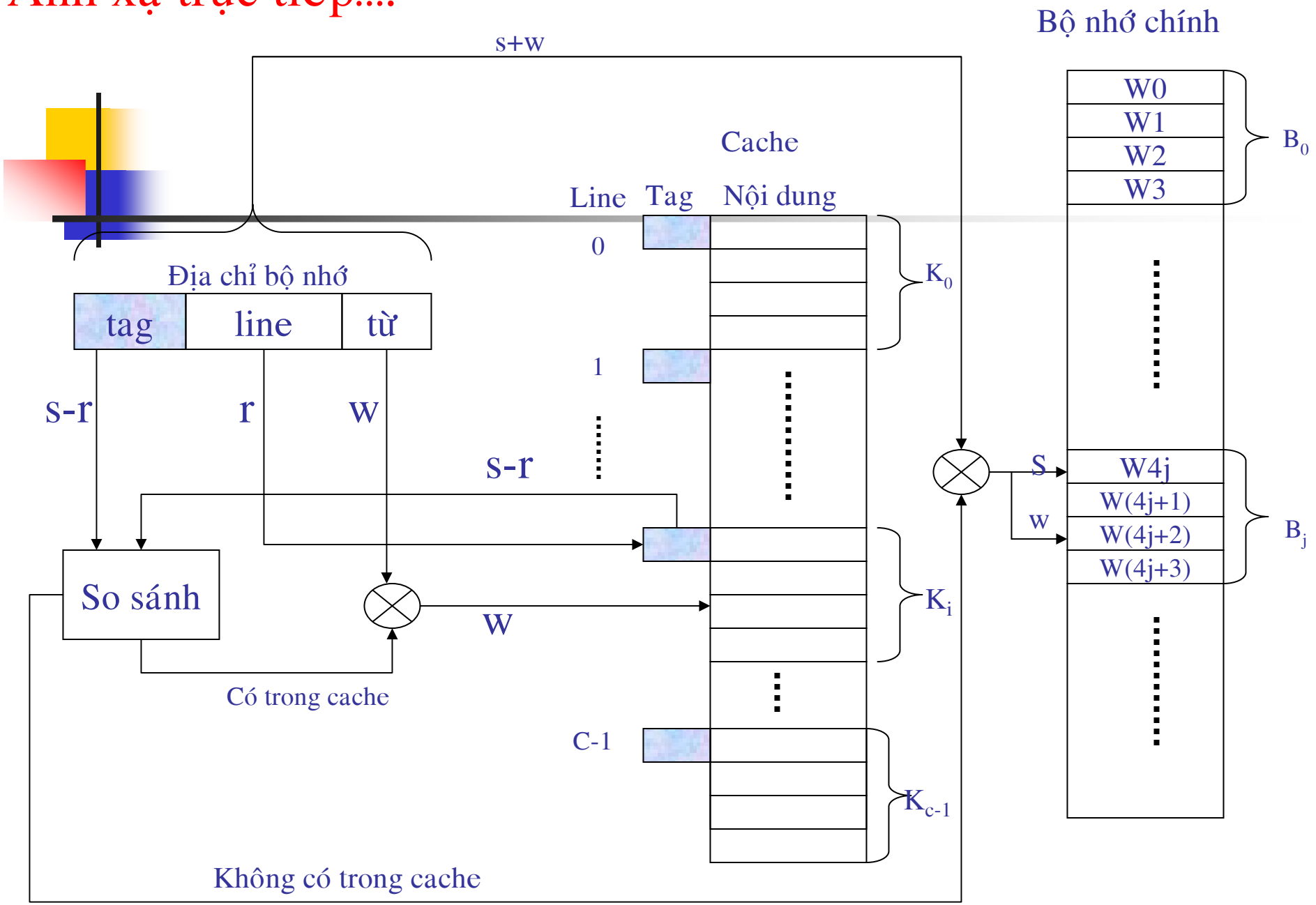
- Tổ chức cache phụ thuộc vào phương pháp ánh xạ được dùng
- Có ba phương pháp ánh xạ chủ yếu
  - Trực tiếp
  - Liên kết đơn
  - Liên kết nhóm

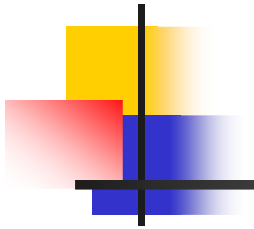


Ánh xạ trực tiếp

- i= j modulo c
  - i : chỉ số line
  - j : chỉ số khối bộ nhớ chính
  - c : số lượng line của cache
- Hàm ánh xạ dễ dàng thực hiện bằng địa chỉ. Để phục vụ cho việc truy xuất, mỗi địa chỉ bộ nhớ chính có thể được chia thành ba phần:
  - w bit thấp nhất định danh cho 1 từ hay một byte trong một khối
  - s bit còn lại chỉ ra một trong 2^s khối bộ nhớ chính
    - Trong đó s-r bit cao nhất là danh định của tag
    - r bit còn lại là danh định của cache line.

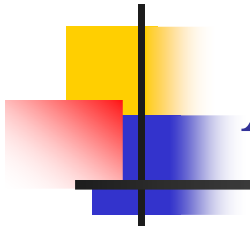
Ánh xạ trực tiếp....





Ánh xạ trực tiếp.....

Cache line	Các khối bộ nhớ được gán
0	$0,c,...,2^s-c$
1	$1,c+1,...,2^s-c+1$
⋮	⋮
c-1	$c-1,2c-1,....,2^s-1$



Ánh xạ trực tiếp...

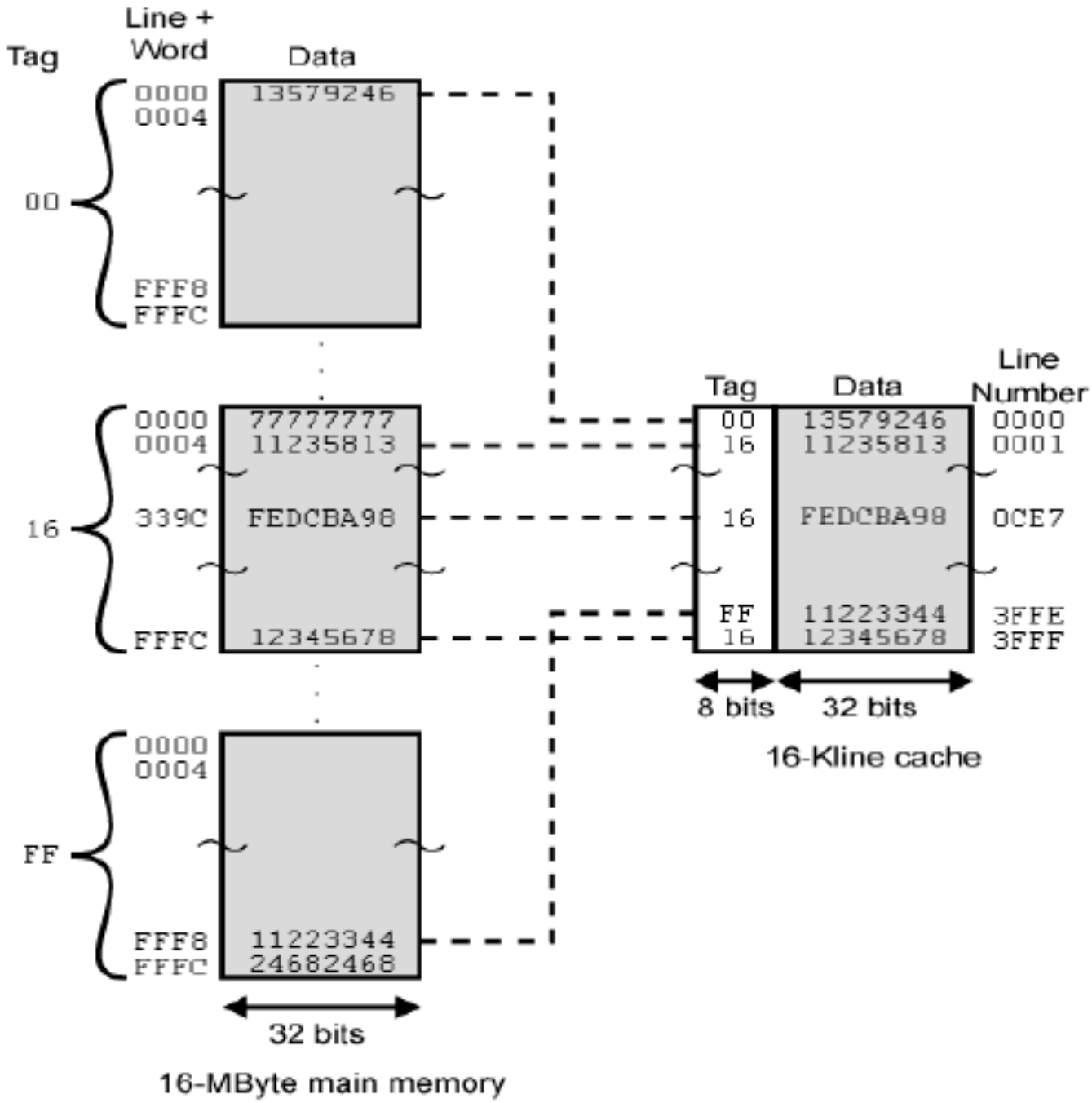
- Đơn giản, chi phí thực hiện thấp
- Một khối bộ nhớ cho trước chỉ được gán một vị trí cache cố định => hệ số tìm thấy sẽ thấp trong một số trường hợp, không tận dụng tối đa hiệu suất cache

Ánh xạ trực tiếp...

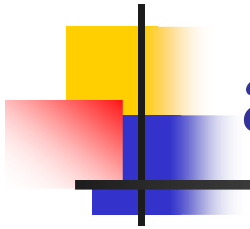
- Ví dụ
  - Kích thước cache 64Kbyte
  - K=4byte
  - Bộ nhớ chính 16Mbyte=2^{24} (24 đường địa chỉ)

  ⇒ M=2^{22} Block, C=2^{14}

Trình bày hoạt động ánh xạ và cho biết các khối bộ nhớ chính nào vào line nào của cache?

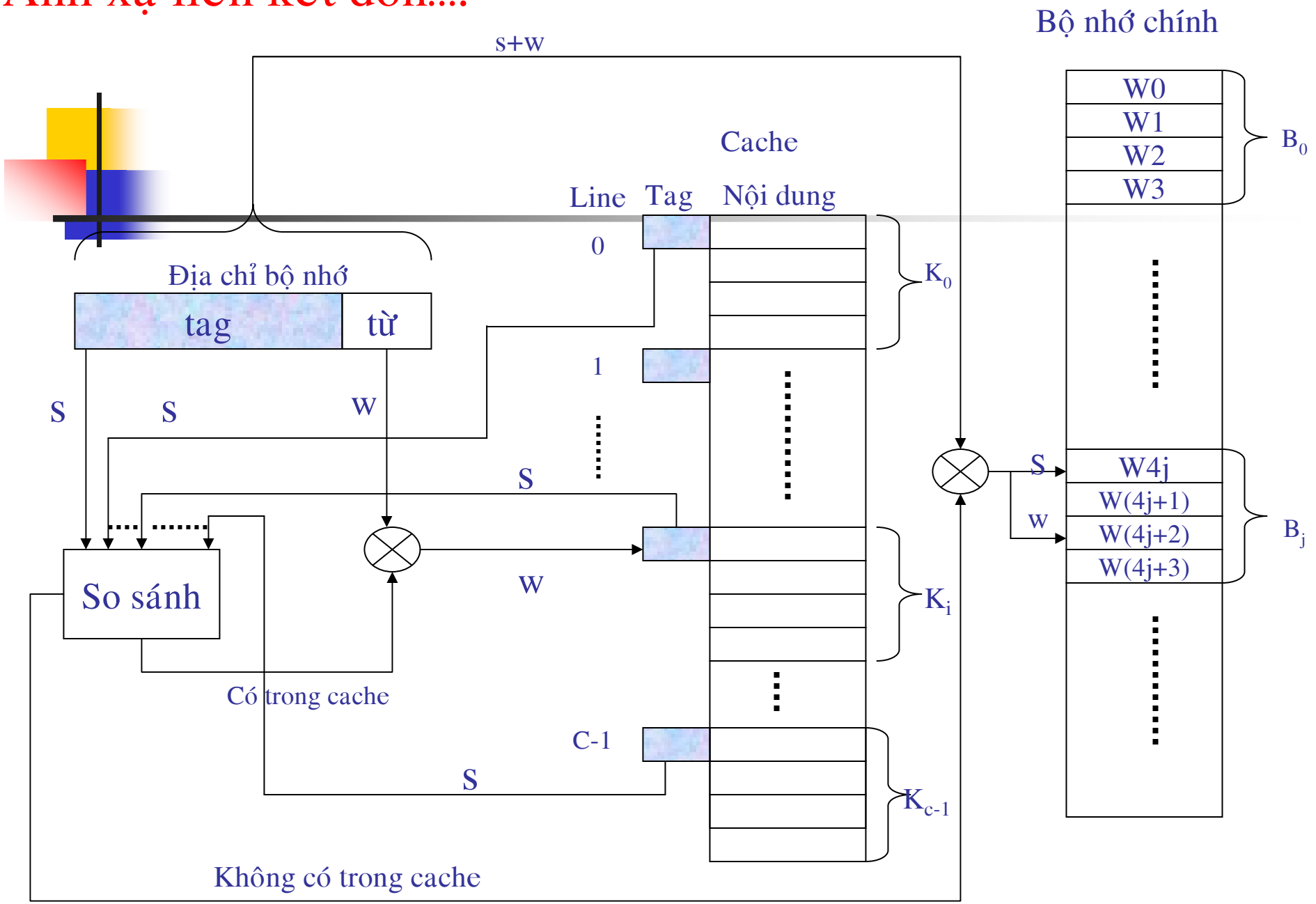

Tag
Line + Word
Data
0000
0004
13579246
00
FFF8
FFFC
0000
0004
77777777
11235813
16
339C
FEDCBA98
FFFC
12345678
0000
0004
FF
FFF8
FFFC
11223344
24682468
32 bits
16-MByte main memory
Tag
Data
Line Number
00
16
13579246
11235813
0000
0001
16
FEDCBA98
0CE7
FF
16
11223344
12345678
3FFE
3FFF
8 bits
32 bits
16-Kline cache

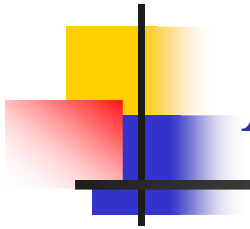
Ánh xạ liên kết đơn (full associative mapping)



- Cho phép mỗi khối bộ nhớ được ánh xạ vào bất kỳ line nào của cache.
- Địa chỉ bộ nhớ gồm có hai phần:
  - Tag: định danh duy nhất cho một khối bộ nhớ
  - Từ: vị trí nội dung cần lấy trong khối bộ nhớ

Ánh xạ liên kết đơn....

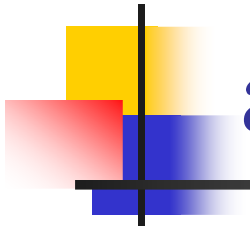




Ánh xạ liên kết đơn...

- Để xác định khối có trong cache hay không, logic điều khiển cache phải kiểm tra tag ở mọi line => mạch thực hiện kiểm tra khá phức tạp.
- Linh hoạt chọn block để thay thế khi đọc một block mới vào cache => thiết kế các thuật toán thay thế để tối đa hệ số tìm thấy

Ánh xạ liên kết nhóm (set associative mapping)



- Dung hòa ưu điểm của cả hai phương pháp trên.
- Toàn bộ cache được chia thành v nhóm, mỗi nhóm có k line.
- Phép ánh xạ như sau:

$$c = v.k$$

$$i = j \text{ modulo } v$$

i: chỉ số của nhóm trong cache

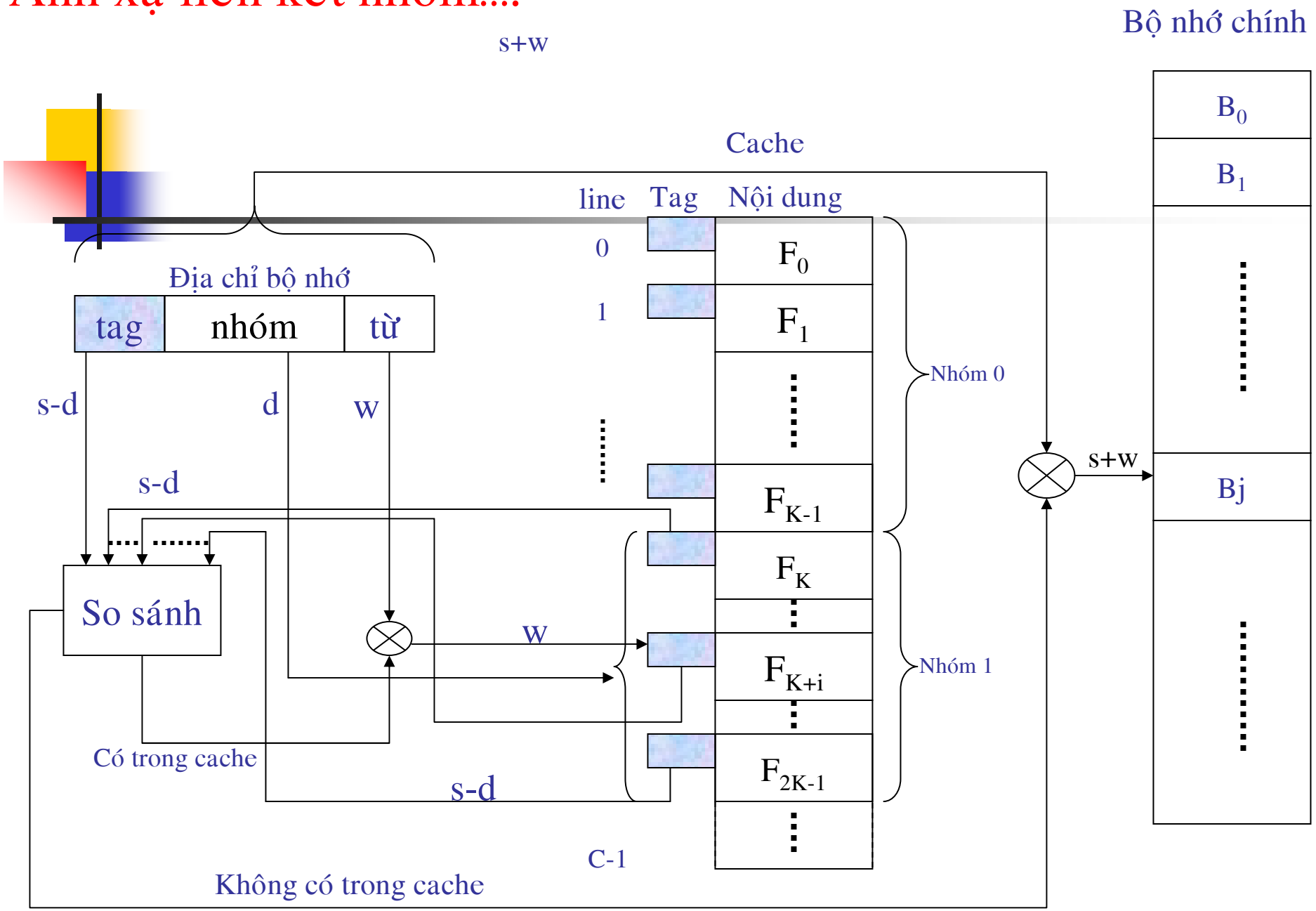
j: chỉ số của khối bộ nhớ chính

c: tổng số line trong cache

Ánh xạ liên kết nhóm (tt)

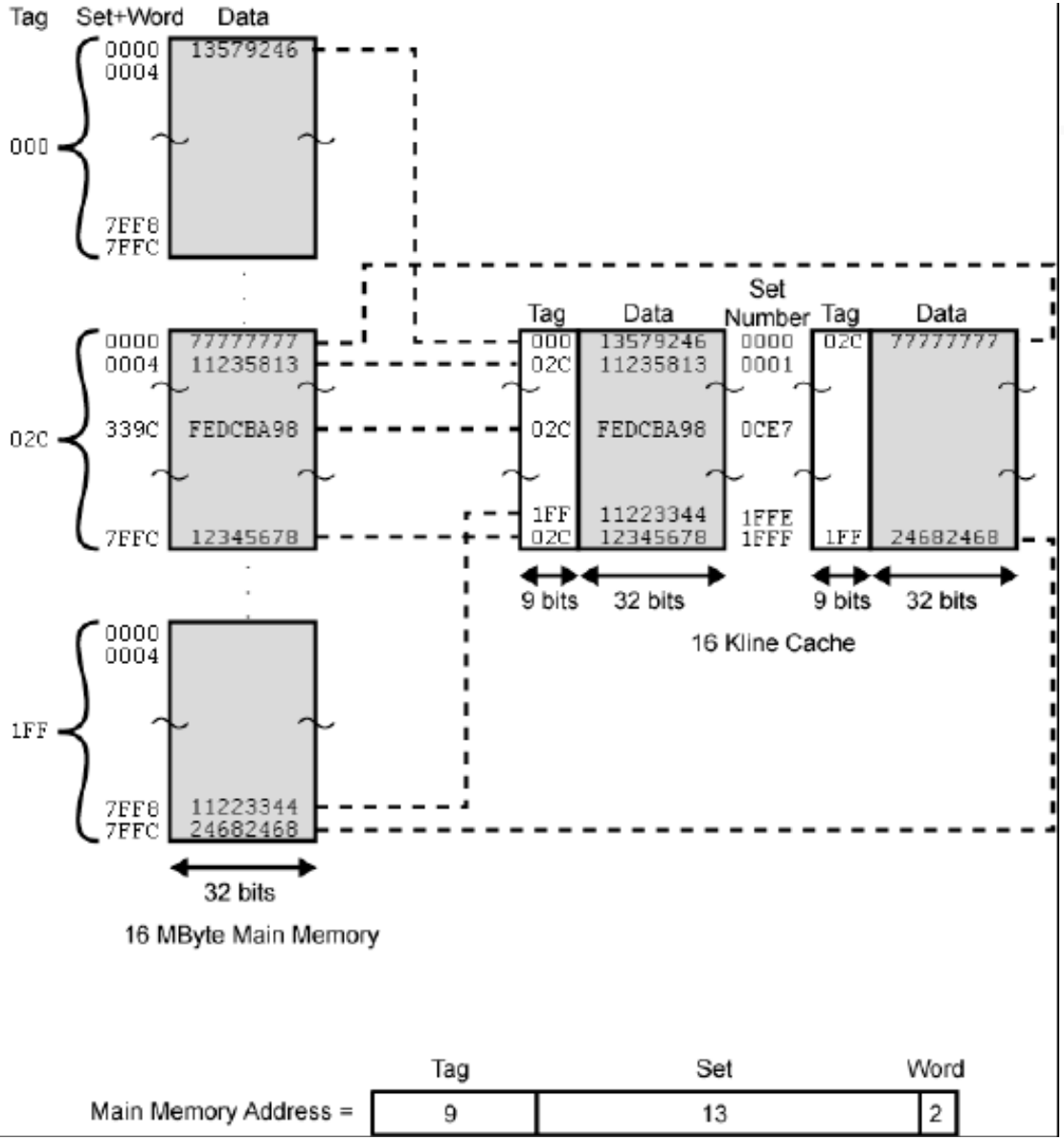
- Khối Bj được ánh xạ vào bất kỳ line nào trong nhóm i
- Địa chỉ bộ nhớ có ba phần:
  - Tag
  - Nhóm
  - Từ
- d bit chỉ ra có 2^d nhóm, s-d là số bit của phần tag. Như vậy bộ nhớ chính được chia thành 2^s khối.

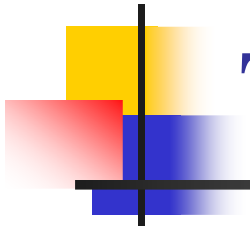
Ánh xạ liên kết nhóm....



Ví dụ 2 line/nhóm
(two way associative mapping)

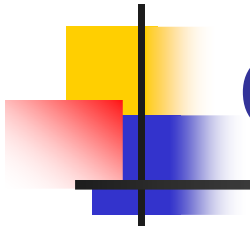
Two Way





Thuật toán thay thế

- Least-recently used
- FIFO
- Least-frequently used
- Random

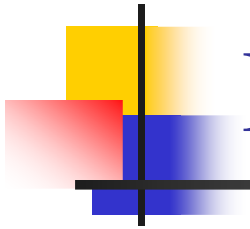


Chính sách ghi

- Hai vấn đề
  - Thay đổi một từ trong cache → từ trong bộ nhớ không hợp lệ
  - Thay đổi một từ trong bộ nhớ→ từ trong cache không hợp lệ
- Kỹ thuật ghi
  - Write througth
  - Write back : dùng UPDATE bit

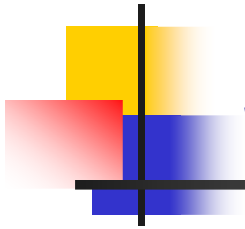
Chính sách ghi...

- Tổ chức bus với nhiều thiết bị (bộ xử lý) có cache riêng và dùng chung một main memory→ vấn đề mới
- Cache coherency system
  - Bus watching with Write Througth: mỗi bộ điều khiển cache đều giám sát các đường địa chỉ để phát hiện hoạt động bộ nhớ với các bus master khác, nếu có ghi vào một vị trí bộ nhớ nào đó mà một line cache có chứa thì bộ điều khiển sẽ đặt line vào trạng thái bất hợp lệ.
  - Hardware Transparency: bổ sung phần cứng để đảm bảo mọi cập nhật bộ nhớ đều được thông báo với tất cả các cache
  - No-cachable memory: chỉ một phần bộ nhớ được chia sẻ cho nhiều bộ xử lý (thiết bị). Không bao giờ copy phần bộ nhớ chia sẻ vào trong cache.



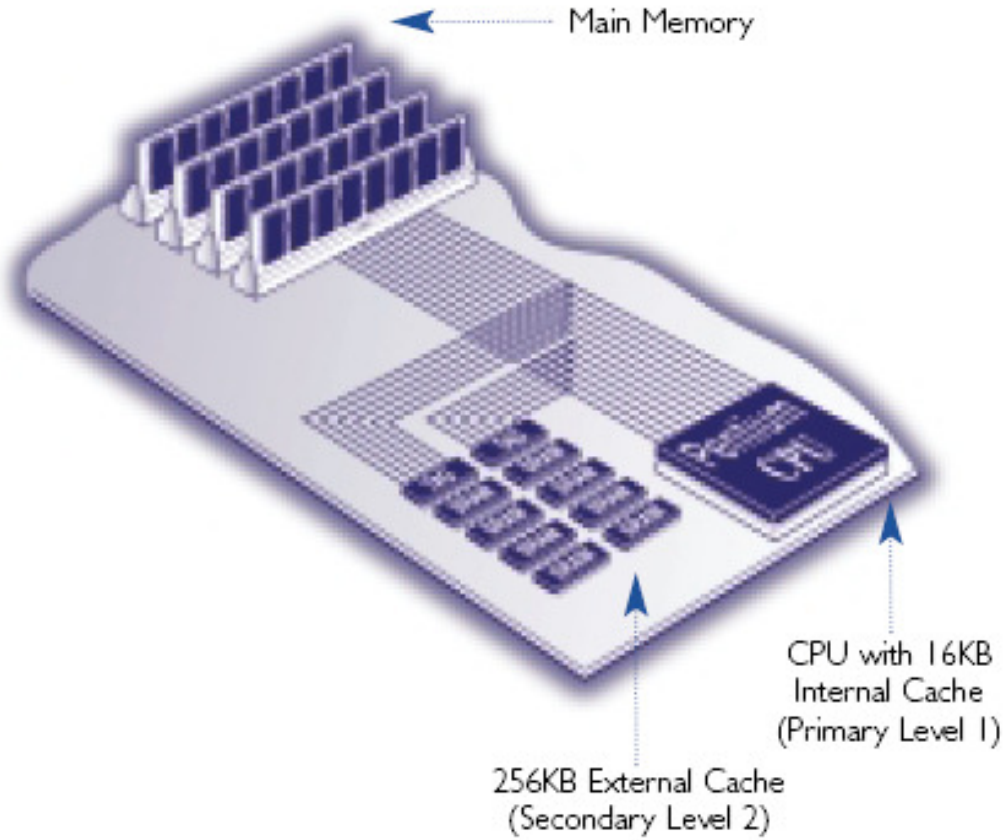
Kích thước khối

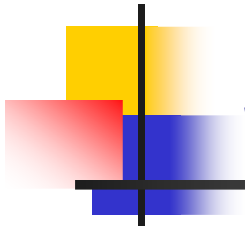
- Khi một khối bắt đầu gia tăng kích thước từ nhỏ đến lớn. Thoạt đầu hit ratio tăng, khi kích thước khối lớn đến một mức nào đó hit ratio lại giảm.
- Khối lớn→ số line giảm
- Khi khối lớn →mỗi từ thêm vào trở nên xa với từ đang tham chiếu và hiếm khi được tham chiếu trong tương lai gần.
- Quan hệ giữa kích thước khối và hit ratio rất phức tạp, khó tìm ra chính xác quan hệ nào trong đó hit ratio luôn tối ưu.



Số lượng cache

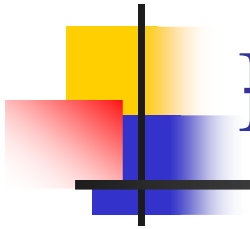
- Cache đơn
  - On-chip cache (internal cache): Cache nằm trên cùng chip với CPU.
  - Off-chip cache (external cache) : Cache có thể đọc qua external bus
- Cache hai mức: dùng cả hai loại
  - internal cache_L1
  - và external cache _L2 (thường dùng SRAM)


Main Memory
CPU with 16KB
Internal Cache
(Primary Level 1)
256KB External Cache
(Secondary Level 2)



Số lượng cache...

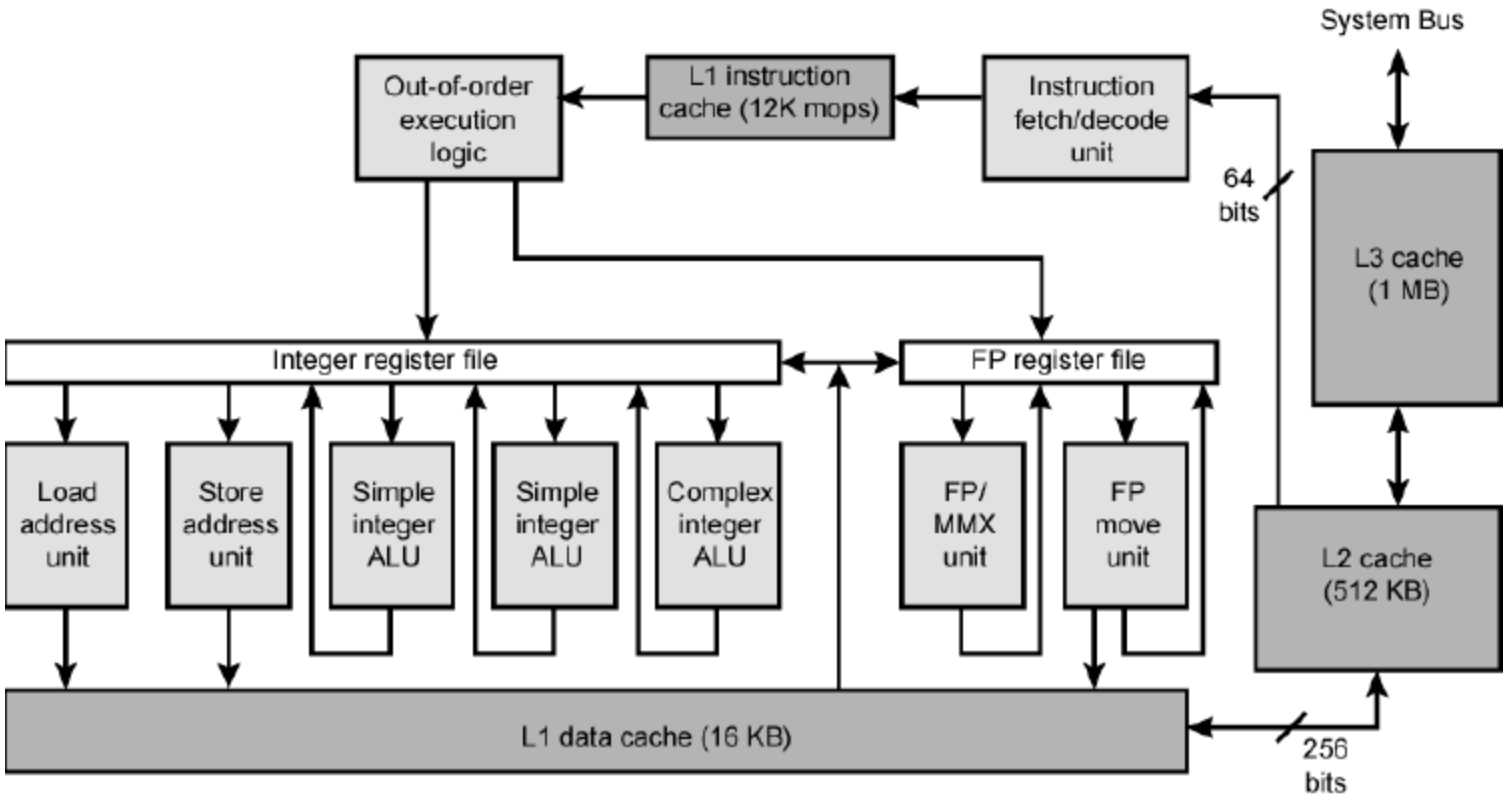
- Hợp nhất
  - Khi mới xuất hiện on-chip cache, nhiều thiết kế dùng một cache để chứa data và instruction
  - Có hit ratio cao, do có điều kiện làm việc theo Data-bound hay Instruction-bound
  - Đơn giản cho thiết kế và hiện thực
- Tách biệt
  - Những thiết kế gần đây chia cache thành hai phần: một chứa data và một chứa insruction
  - Hỗ trợ xử lý song song, pipelining các chỉ thị
  - Ngăn chặn sự tranh chấp giữa bộ xử lý chỉ thị và đơn vị thực thi.



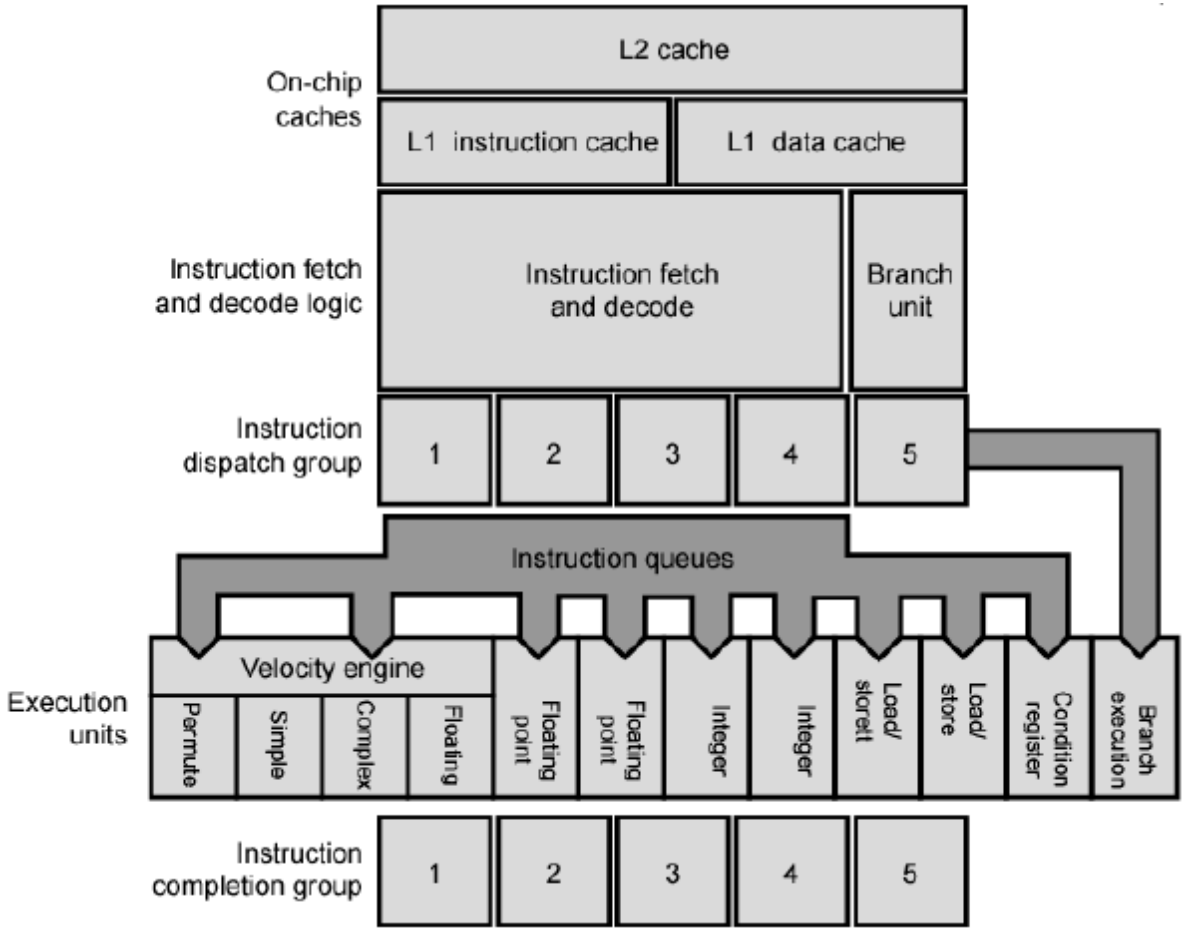
Điều khiển cache

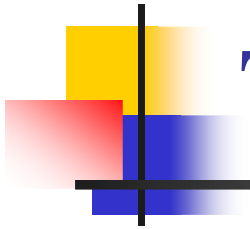
- Cache bên trong được điều khiển bởi hai bit của một thanh ghi điều khiển, gọi là CD (cache disable) và NW (not write-througth)
- Có hai chỉ thị
  - INVD: hủy bỏ nội dung trong cache và báo cho cache ngoài.
  - WBINVD: cũng có chức năng tương tự nhưng báo cho cache ngoài thực hiện ghi vào bộ nhớ khối đã hiệu chỉnh.

Sơ đồ cache trong Pentium IV



Sơ đồ cache trong PowerPC G5





Tính toán với cache

- Hiệu suất cache được đánh giá qua:

CPU execution time =(CPU clock cycles + Memory stall cycles)× Clock cycle time

- Giả sử CPU clock cycles đã bao gồm thời gian xử lý một cache hit và CPU bị treo trong khi cache miss.

Tính toán với cache (tt)

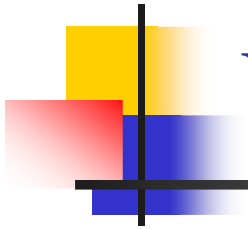
- Số chu kỳ treo do đợi bộ nhớ là:

$$\text{Memory stall cycles} = \text{Number of misses} \times \text{Miss penalty}$$

$$= \text{IC} \times \frac{\text{Misses}}{\text{Instruction}} \times \text{Miss penalty}$$

$$= \text{IC} \times \frac{\text{Memory accesses}}{\text{Instruction}} \times \text{Miss rate} \times \text{Miss penalty}$$

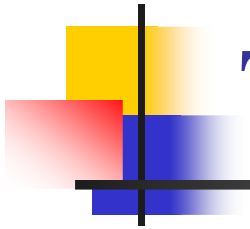
- Average memory access time = Hit time+Miss rate.Miss penalty



Ví dụ

Một máy tính chỉ cần 1 clock cho một chỉ thị nếu tìm thấy trong cache. Số lần truy xuất data chiếm 50%. Chi phí không tìm thấy trong cache là 25 chu kỳ clock và hệ số hit là 98%. Nếu tất cả các chỉ thị đều tìm thấy trong cache thì máy tính chạy nhanh gấp bao nhiêu lần so với trường hợp này?

Giải

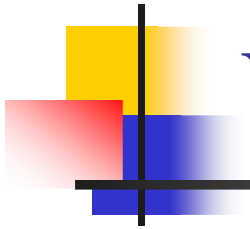


Tính số hit trong hoạt động ghi

Có hai tùy chọn với hoạt động ghi:

-Write allocate: Cấp block trên cache cho hoạt động ghi.

-No-write allocate: Không cấp block, block được cập nhật ngay trên bộ nhớ.



Ví dụ

Giả sử ánh xạ cache là liên kết với write-back. Xem xét đoạn chương trình sau:

write M[100]

write M[100]

read M[200]

write M[200]

write M[100]

Hãy tính số hit và miss:

a, với write allocate

b, với no-write allocate