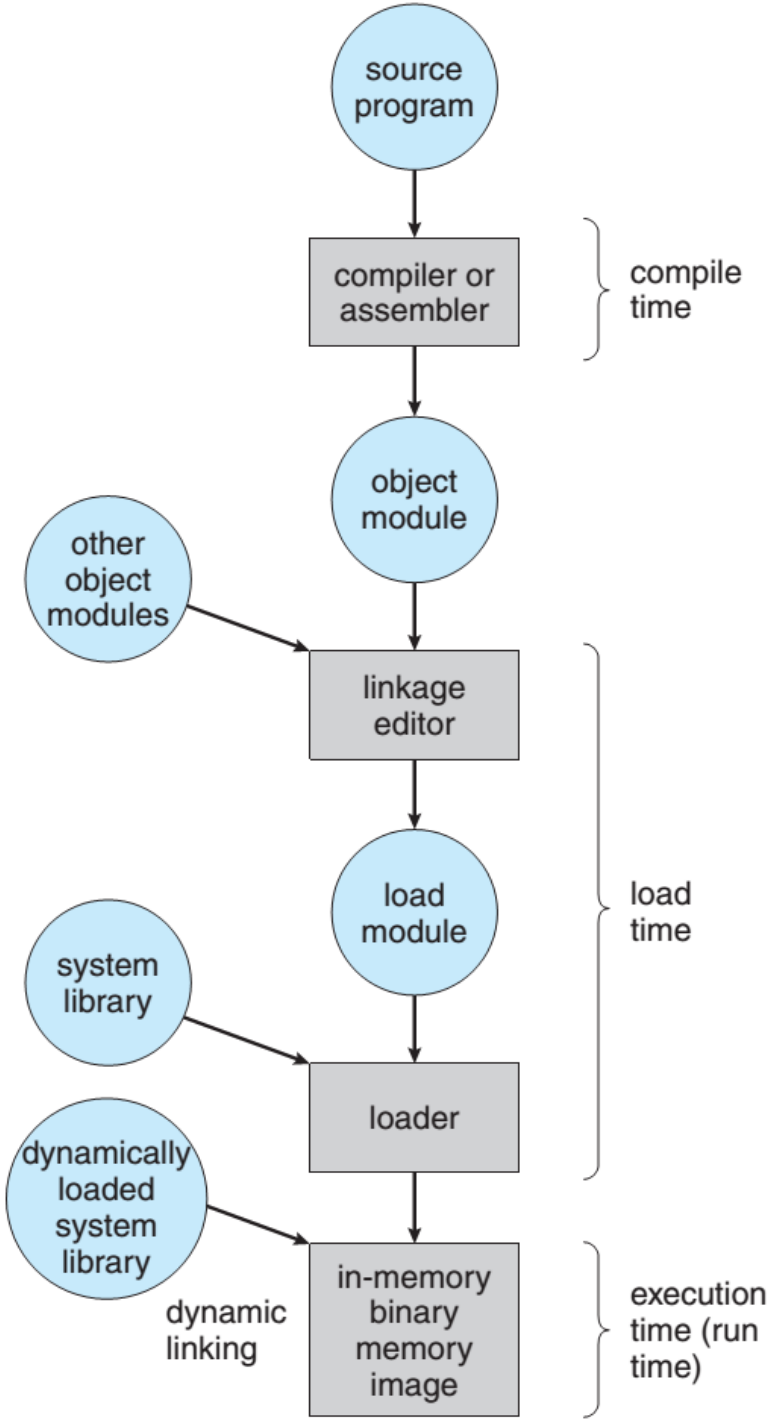
QUẢN LÝ BỘ NHỚ

GV. Nguyễn Thị Hải Bình

Khoa CNTT, ĐH Giao thông vận tải

Email: calmseahn@gmail.com

Website: calmseahn.weebly.com



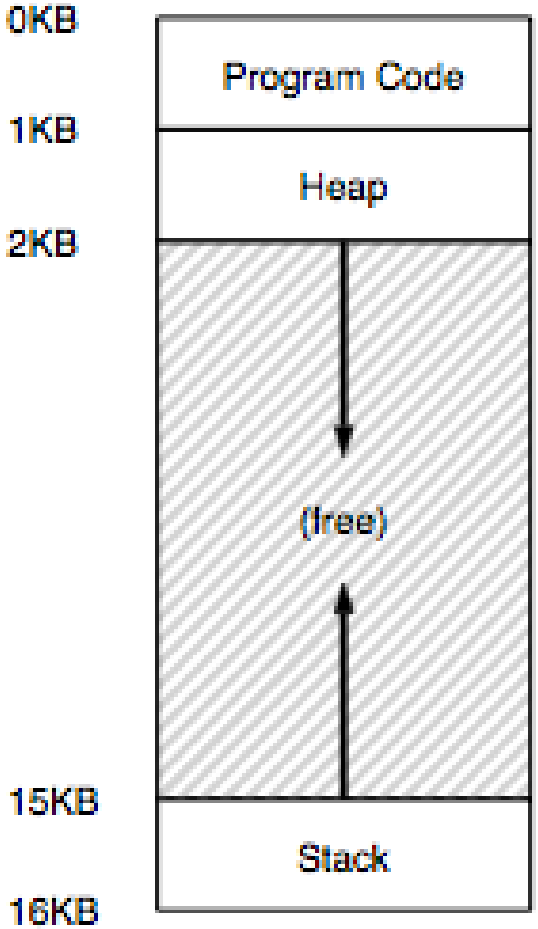
Multistep processing of a user program

KẾT BUỘC ĐỊA CHỈ (ADDRESS BINDING)

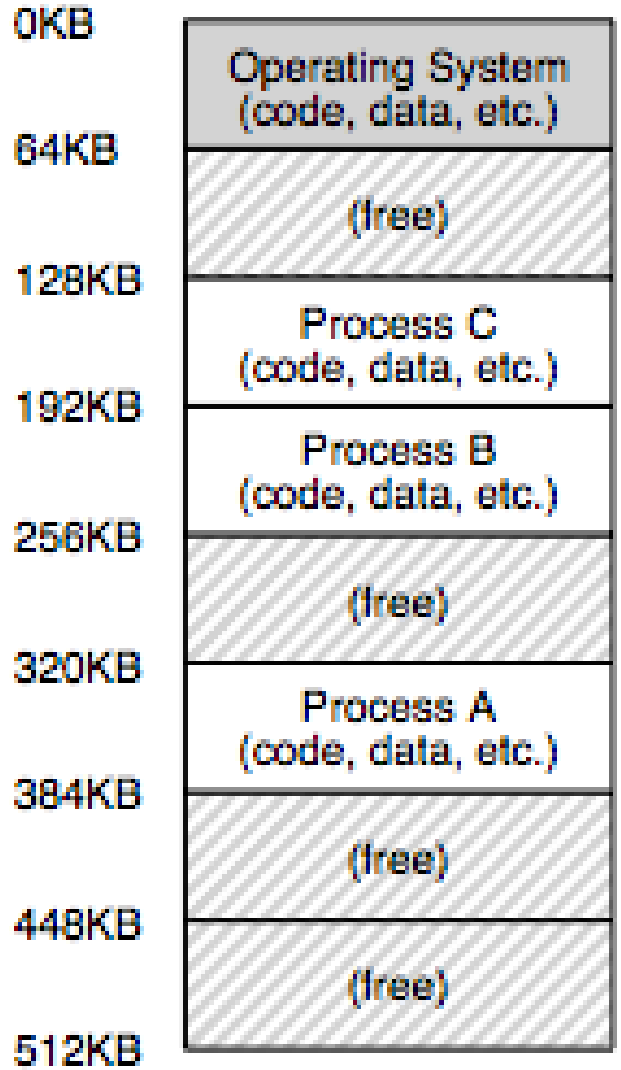
- Quá trình kết buộc các chỉ thị và dữ liệu của chương trình với địa chỉ cụ thể trong bộ nhớ có thể được thực hiện tại một trong các thời điểm sau:
  - Thời điểm biên dịch
    - Nếu tại thời điểm biên dịch biết được tiến trình sẽ nằm đầu trong bộ nhớ, trình biên dịch có thể sinh mã với địa chỉ tuyệt đối
    - Nếu cần thay đổi địa chỉ thì phải biên dịch lại
  - Thời điểm tải
    - Trình biên dịch sinh mã với địa chỉ có thể định vị lại
    - Nếu cần thay đổi địa chỉ, chỉ cần tải lại chương trình
  - Thời điểm thực thi
    - Được sử dụng trong trường hợp tiến trình có thể di chuyển từ vùng nhớ này sang vùng nhớ khác
    - Đòi hỏi phần cứng đặc biệt

KHÔNG GIAN ĐỊA CHỈ LOGIC VÀ ĐỊA CHỈ VẬT LÝ

- Địa chỉ logic (logical address):
  - Sinh bởi CPU. Còn gọi là địa chỉ ảo (Virtual address)
  - Cấp phát cho các biến khi biên dịch chương trình
- Địa chỉ vật lý (physical address):
  - Là địa chỉ cụ thể trong bộ nhớ
  - Được cấp phát cho các biến khi thực hiện chương trình
- Địa chỉ logic và vật lý giống nhau trong trường hợp kết buộc địa chỉ tại thời điểm biên dịch và thời điểm tải; khác nhau trong trường hợp kết buộc tại thời điểm thực thi.
- Kết buộc địa chỉ (binding) là ánh xạ (mapping) từ không gian địa chỉ này sang không gian địa chỉ khác.



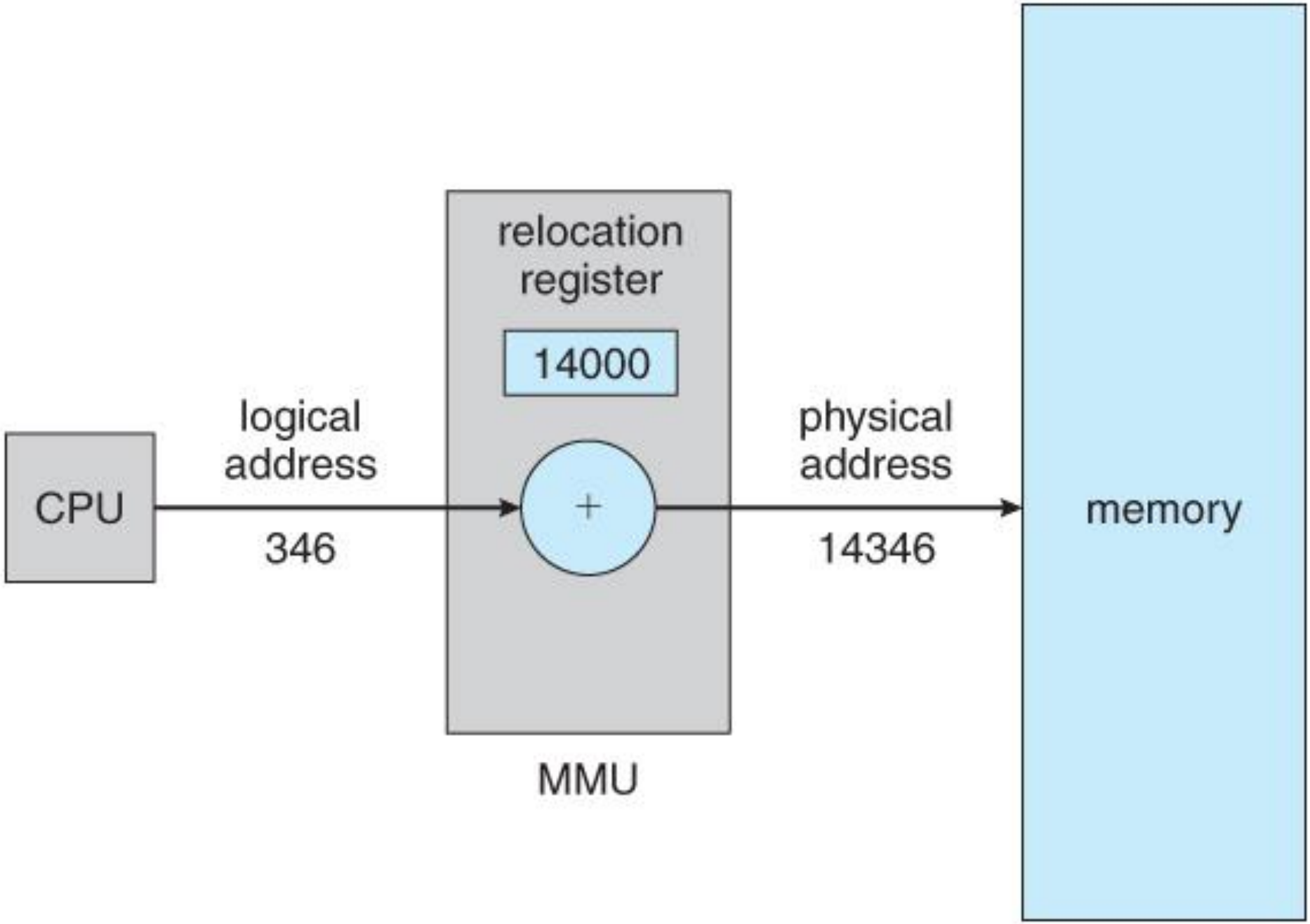
Logical address space for process A



Three process sharing the physical address space

ĐƠN VỊ QUẢN LÝ BỘ NHỚ

- Thuật ngữ: Memory Management Unit (MMU)
- Là thiết bị phần cứng dùng để ánh xạ địa chỉ ảo sang địa chỉ vật lý.
- Trong MMU, có thanh ghi relocation (thanh ghi định vị lại) để tính toán địa chỉ vật lý từ địa chỉ ảo.
- Chương trình của người dùng làm việc trên địa chỉ ảo và không bao giờ biết địa chỉ vật lý.


relocation
register
14000
logical
address
346
CPU
+
physical
address
14346
memory
MMU

CÁC CẤU TRÚC CƠ BẢN CỦA CHƯƠNG TRÌNH

- Có nhiều phương pháp tổ chức chương trình ở bộ nhớ trong để thực hiện
- Các phương pháp này khác nhau ở *kiểu định vị chương trình trong bộ nhớ* và *thời điểm thực hiện ánh xạ địa chỉ tương đối thành địa chỉ tuyệt đối*
- Cấu trúc của chương trình thể hiện cách *quản lý bộ nhớ logic* và *hình ảnh của nó ở bộ nhớ vật lý* khi thực hiện
- Các dạng cấu trúc gồm: Cấu trúc tuyến tính, cấu trúc động, Overlay, phân đoạn, phân trang

CÁC CẤU TRÚC CƠ BẢN CỦA CHƯƠNG TRÌNH

- Cấu trúc tuyến tính
- Cấu trúc động
- Cấu trúc overlay
- Cấu trúc phân đoạn
- Cấu trúc phân trang

CẤU TRÚC TUYẾN TÍNH

- Là cấu trúc mà sau khi biên dịch, các modul được tập hợp thành một chương trình hoàn thiện (một modul duy nhất), chứa đầy đủ mọi thông tin để chương trình có thể thực hiện (trừ dữ liệu vào).
- Mọi biến ngoài đều được gán địa chỉ cụ thể
- Khi thực hiện chỉ cần định vị chương trình một lần vào bộ nhớ

CẤU TRÚC TUYẾN TÍNH

- Ưu điểm
  - Đơn giản, dễ tổ chức biên dịch và định vị
  - Thời gian thực hiện nhanh
  - Có tính lưu động cao
- Nhược điểm
  - Lãng phí bộ nhớ

CẤU TRÚC ĐỘNG

- Là cấu trúc mà các modul được biên tập một cách riêng biệt
- Khi thực hiện chương trình, hệ thống chỉ định vị modul gốc
- Trong quá trình thực hiện nếu một modul được gọi tới thì
  - Hệ thống cấp phát không gian nhớ và nạp modul
  - Khi hoạt động xong thì giải phóng modul, thu hồi không gian nhớ

CẤU TRÚC ĐỘNG

M0		
M0	M1	M2
M0	M3	M4
M0	M5	

CẤU TRÚC ĐỘNG

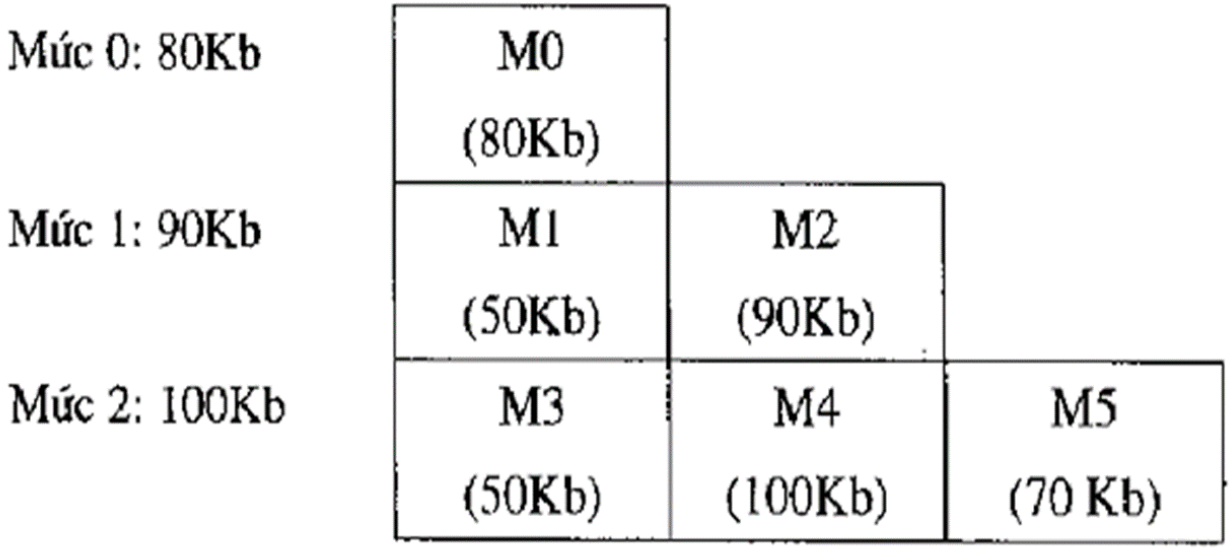
- Ưu điểm
  - Tiết kiệm bộ nhớ
- Nhược điểm
  - Nạp và xoá các modul được thực hiện bởi người dùng
  - Kích thước chương trình lớn
  - Người dùng phải nắm rõ cấu trúc chương trình và các công cụ điều khiển bộ nhớ của hệ điều hành

CẤU TRÚC OVERLAY

- Các modul chương trình sau khi biên dịch được chia thành các mức
  - Mức 0: Chứa modul gốc để nạp chương trình
  - Mức 1: Chứa các modul được gọi bởi mức 0
  - Mức 2: Chứa các modul được gọi bởi mức 1
  - …
- Modul trong cùng một mức không được gọi lẫn nhau

CẤU TRÚC OVERLAY

- Bộ nhớ dành cho chương trình cũng được chia thành các mức tương ứng với các mức chương trình
- Kích thước mỗi mức trong bộ nhớ bằng kích thước modul lớn nhất của mức chương trình tương ứng



CẤU TRÚC OVERLAY

- Để tạo thành chương trình overlay, người sử dụng cần cần cung cấp thông tin về các mức cho trình biên dịch thông qua sơ đồ overlay
- Khi thực hiện chương trình
  - Modul gốc được định vị vào bộ nhớ như chương trình có cấu trúc tuyến tính
  - Cần tới modul nào, hệ thống tìm kiếm trong sơ đồ overlay và nạp vào bộ nhớ ở mức tương ứng

CẤU TRÚC OVERLAY

- Ưu điểm
  - Cho phép sử dụng bộ nhớ nhiều hơn phần bộ nhớ mà hệ thống dành cho chương trình
- Nhược điểm
  - Yêu cầu người dùng cung cấp thông tin về sơ đồ overlay
  - Hiệu quả tiết kiệm bộ nhớ phụ thuộc vào cách tổ chức và bố trí modul của chương trình

CẤU TRÚC PHÂN ĐOẠN

- Chương trình được phân đoạn thành các modul độc lập
- Thông tin về các modul được chứa trong một bảng điều khiển gọi là bảng quản lý đoạn
- Trong bảng quản lý đoạn còn chứa các thông tin trợ giúp việc định vị các modul vào bộ nhớ

CẤU TRÚC PHÂN ĐOẠN

- Ưu điểm
  - Không yêu cầu người sử dụng khai báo thêm thông tin
- Nhược điểm
  - Hiệu quả sử dụng bộ nhớ phụ thuộc vào cách phân chia chương trình thành các modul độc lập
  - Chương trình có cấu trúc phân đoạn chỉ áp dụng được khi bộ nhớ quản lý theo kiểu phân đoạn

CẤU TRÚC PHÂN TRANG

- Chương trình được biên dịch như cấu trúc tuyến tính, sau đó được phân chia thành các phần bằng nhau gọi là trang
- Thông tin về các trang được chứa trong một bảng điều khiển gọi là bảng quản lý trang
- Mỗi phần tử trong bảng quản lý trang tương ứng với một trang trong chương trình của người sử dụng

CẤU TRÚC PHÂN TRANG

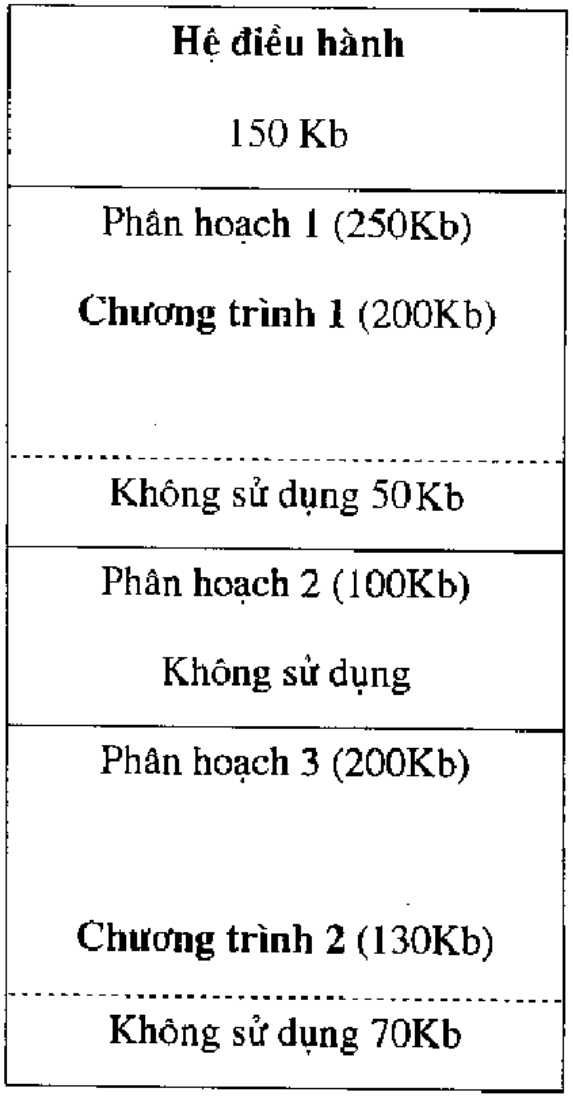
- Ưu điểm
  - Phát huy được hiệu quả sử dụng bộ nhớ
- Nhược điểm
  - Chỉ áp dụng đối với bộ nhớ được quản lý theo kiểu phân trang

CÁC SƠ ĐỒ QUẢN LÝ BỘ NHỚ

- Sơ đồ phân hoạch cố định
- Sơ đồ phân hoạch động
- Sơ đồ hoán đổi
- Sơ đồ phân đoạn
- Sơ đồ phân trang
- Sơ đồ kết hợp phân trang và phân đoạn

SƠ ĐỒ PHÂN HOẠCH CỐ ĐỊNH

- Bộ nhớ được chia thành n phần, mỗi phần được sử dụng như một bộ nhớ độc lập
- Mỗi phần đó gọi là một phân hoạch



SƠ ĐỒ PHÂN HOẠCH CỐ ĐỊNH

- Các thuật toán lựa chọn phân hoạch để sử dụng
  - First Fit: Chọn phân hoạch đầu tiên đủ lớn để cấp phát
  - Best Fit: Chọn phân hoạch nhỏ nhất đủ để cấp phát
  - Worst Fit: Chọn phân hoạch lớn nhất để cấp phát
- Ưu điểm:
  - Đơn giản, dễ tổ chức
- Nhược điểm
  - Chương trình sẽ không thực hiện nếu nó có kích thước lớn hơn phân hoạch lớn nhất
  - Hiện tượng phân mảnh: không gian bộ nhớ trống bị phân thành nhiều mảnh nhỏ
    - Phân mảnh ngoài: Tổng lượng bộ nhớ trống đủ lớn để đáp ứng một yêu cầu nào đó, nhưng các khoảng trống nằm không liên tục trên toàn bộ nhớ
    - Phân mảnh trong: Kích thước của phân hoạch lớn hơn kích thước chương trình → Không gian trống bên trong mỗi phân hoạch

SƠ ĐỒ PHÂN HOẠCH ĐỘNG

- Số lượng phân hoạch trên bộ nhớ và kích thước của mỗi phân hoạch không cố định
- Khi một tiến trình được nạp vào bộ nhớ, hệ điều hành cấp phát cho nó một không gian nhớ vừa đủ để chứa tiến trình
- Ưu điểm
  - Không gây phân mảnh trong
- Nhược điểm
  - Phân mảnh ngoài

SƠ ĐỒ PHÂN HOẠCH ĐỘNG

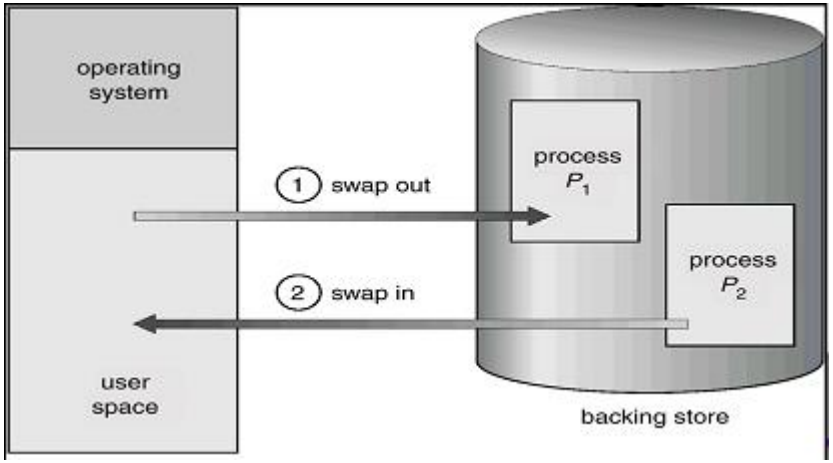
- Khắc phục hiện tượng phân mảnh ngoài
  - Tìm thời điểm thích hợp để dừng tiến trình
  - Đưa một số hoặc tất cả các tiến trình ra ngoài bộ nhớ (swap)
  - Tái định vị các tiến trình và khôi phục trạng thái hoạt động

<table>
<tr><td>Hệ điều hành
150KB</td><td></td><td>Hệ điều hành
150KB</td><td></td><td>Hệ điều hành
150KB</td></tr>
<tr><td>Tiến trình 1
40KB</td><td></td><td rowspan="2">Bộ nhớ tự do
120KB</td><td></td><td>Tiến trình 5
30KB</td></tr>
<tr><td>Tiến trình 2
80KB</td><td></td><td></td><td>Bộ nhớ tự do
90KB</td></tr>
<tr><td>Tiến trình 3
70KB</td><td></td><td>Tiến trình 3
70KB</td><td></td><td>Tiến trình 3
70KB</td></tr>
<tr><td>Tiến trình 4
40KB</td><td></td><td>Tiến trình 4
40KB</td><td></td><td>Bộ nhớ tự do
40KB</td></tr>
</table>

<table>
<tr><td>Hệ điều hành
150KB</td><td></td><td>Hệ điều hành
150KB</td></tr>
<tr><td>Tiến trình 5
30KB</td><td></td><td>Tiến trình 5
30KB</td></tr>
<tr><td rowspan="3">Tổ chức lại
200KB</td><td></td><td>Tiến trình 3
70KB</td></tr>
<tr><td></td><td rowspan="2">Bộ nhớ tự do
130KB</td></tr>
<tr><td></td></tr>
</table>

SƠ ĐỒ HOÁN ĐỔI (SWAPPING)

- Nguyên tắc:
  - Tiến trình ở trạng thái chờ trong thời gian tương đối dài được chuyển ra bộ nhớ ngoài để giải phóng vùng nhớ
  - Khi tiến trình kết thúc chờ, nó được nạp vào bộ nhớ để tiếp tục thực hiện
- Cần giải quyết vấn đề phân mảnh ngoài



SƠ ĐỒ PHÂN ĐOẠN

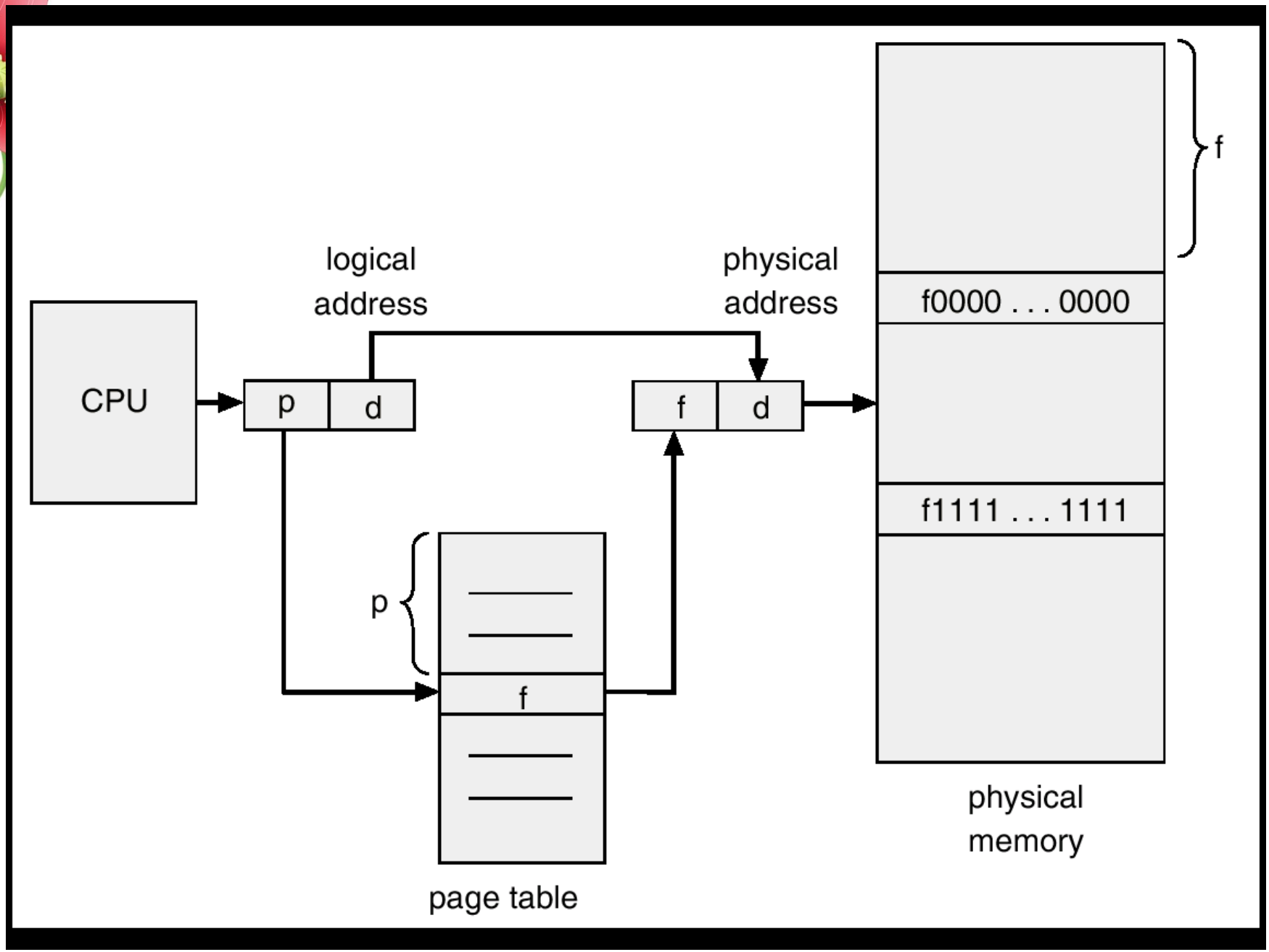
- Các modul chương trình được biên dịch một cách riêng biệt
- Thông tin về modul được chứa trong bảng quản lý đoạn
  - Dấu hiệu D: cho biết modul đã được nạp vào bộ nhớ hay chưa (D = 0 tương ứng với chưa được nạp)
  - Địa chỉ A của vùng nhớ được định vị
  - Độ dài L: kích thước của modul

SƠ ĐỒ PHÂN ĐOẠN

- Ưu điểm
  - Không đòi hỏi công cụ tổ chức đặc biệt, do vậy có thể áp dụng trên mọi hệ thống
- Nhược điểm
  - Hiệu quả sử dụng bộ nhớ phụ thuộc vào cấu trúc chương trình của người sử dụng
  - Hiện tượng phân mảnh bộ nhớ

SƠ ĐỒ PHÂN TRANG

- Bộ nhớ vật lý được chia thành các khung trang (frame) có kích thước cố định
- Bộ nhớ logic được chia thành các trang (page)
- Kích thước trang và khung trang bằng nhau
- Trước khi thực thi, các trang của tiến trình nằm trên ổ đĩa sẽ được tải vào bất kì khung trang chưa sử dụng nào của bộ nhớ
- Địa chỉ CPU tạo ra được chia thành hai phần: địa chỉ trang (p) và địa chỉ tương đối trong trang (d)
- Địa chỉ trang được dùng làm chỉ mục đến bảng trang
- Bảng trang lưu trữ địa chỉ cơ sở của mỗi trang trong bộ nhớ vật lý
- Địa chỉ cơ sở cộng với địa chỉ tương đối trong trang tạo ra địa chỉ vật lý


CPU
logical address
p
d
physical address
f
d
p
f
page table
f
f0000 . . . 0000
f1111 . . . 1111
physical memory

SƠ ĐỒ PHÂN TRANG

- Ưu điểm
  - Cho phép không gian địa chỉ logic của tiến trình không nằm liên tục trong bộ nhớ vật lý
  - Khắc phục hiện tượng phân mảnh ngoài
- Nhược điểm
  - Cần có thiết bị vật lý hỗ trợ định vị trang

SƠ ĐỒ PHÂN TRANG

- Các giải pháp nạp trang
  - Nạp tất cả các trang của chương trình vào bộ nhớ ngay từ đầu
  - Giải pháp nạp trước: Cho phép người sử dụng tạo chương trình ở bộ nhớ logic với kích thước tùy ý. Do vậy phải dự báo được các trang cần thiết chuẩn bị sử dụng trong quá trình thực hiện
  - Nạp trang theo yêu cầu: Trang chỉ được nạp khi xuất hiện yêu cầu truy cập dữ liệu của trang

SƠ ĐỒ PHÂN TRANG

- Các giải pháp thay thế trang:
  - FIFO: Trang nạp trước thay thế trước.
  - LRU(Last Recently Used): Thay thế trang có lần sử dụng cuối cách thời điểm đổi trang lâu nhất.
  - LFU(Last Frequently Used): Thay thế trang có tần suất sử dụng thấp nhất
  - Thuật toán tối ưu: Thay thế trang sẽ không được sử dụng trong khoảng thời gian dài nhất

SƠ ĐỒ KẾT HỢP PHÂN TRANG VÀ PHÂN ĐOẠN

- Chương trình được biên dịch theo sơ đồ phân đoạn và có bảng quản lý đoạn chung
- Mỗi đoạn được biên tập theo sơ đồ phân trang
- Mỗi đoạn có bảng quản lý trang riêng

BỘ NHỚ ẢO

- Bộ nhớ ảo (Virtual memory) là một kỹ thuật cho phép xử lý một chương trình không được nạp toàn bộ vào bộ nhớ vật lý
- Bộ nhớ ảo mô hình hóa bộ nhớ như một bảng lưu trữ rất lớn và đồng nhất, tách biệt hẳn với khái niệm bộ nhớ logic và bộ nhớ vật lý
- Người sử dụng chỉ nhìn thấy và làm việc trong không gian địa chỉ ảo, còn việc chuyển đổi sang bộ nhớ vật lý do hệ điều hành thực hiện

CÀI ĐẶT BỘ NHỚ ẢO

- Bộ nhớ ảo có thể được cài đặt dựa vào hai kỹ thuật: Phân trang theo yêu cầu và phân đoạn theo yêu cầu
- Kỹ thuật phân trang theo yêu cầu được áp dụng phổ biến
  - Sử dụng kỹ thuật phân trang kết hợp với kỹ thuật swapping
  - Yêu cầu có một cớ chế phần cứng để phân biệt các trang ở bộ nhớ trong và ngoài
    - Bảng trang: Phản ánh được tình trạng một trang đang ở bộ nhớ trong hay bộ nhớ ngoài.
    - Bộ nhớ ngoài: Lưu trữ các trang không được nạp vào bộ nhớ trong

HIỆN TƯỢNG LỖI TRANG

- Khi hệ thống truy xuất đến một trang, nhưng trang này chưa được nạp vào bộ nhớ trong sẽ làm phát sinh lỗi trang
- Hệ điều hành xử lý lỗi theo các bước sau:
  1. Kiểm tra địa chỉ truy nhập có nằm trong vùng địa chỉ hợp lệ không
  2. Nếu yêu cầu không hợp lệ, kết thúc công việc. Ngược lại, nếu trang yêu cầu chưa nằm trong bộ nhớ thì HĐH tải trang vào bộ nhớ
  3. Tìm một frame trống
  4. Yêu cầu đọc trang mong muốn từ ổ đĩa vào frame mới tìm thấy
  5. Cập nhật bảng trạng thái của tiến trình và bảng phân trang
  6. Khởi động lại chỉ thị bị ngắt do lỗi trang

THAY THẾ TRANG

- Khi hệ thống nhận yêu cầu nạp trang mới vào bộ nhớ, nếu không còn trang trống trong bộ nhớ trong, hệ thống cần thực hiện việc thay thế trang
- Các thuật toán thay thế trang áp dụng như trong sơ đồ quản lý bộ nhớ kiểu phân trang
  - FIFO
  - LRU
  - LFU
  - Thuật toán tối ưu

BÀI TẬP

- Cho chuỗi tham chiếu 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 3, 1, 2, 0, 1, 7, 0, 1
- Bộ nhớ có 3 frame
- Nếu sử dụng thuật toán FIFO thì có mấy lỗi trang
- Nếu sử dụng thuật toán LRU (Least Recently Used) thì có mấy lỗi trang

FIFO

7	**0**	**1**	**2**	**0**	**3**	**0**	**4**	**2**	**3**	**0**	**3**	**2**	**1**	**2**	**0**	**1**	7	**0**	**1**
7	7	7	2	2	2	2	4	4	4	0	0	0	0	0	0	0	7	7	7
	0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1	1	0	0
		1	1	1	1	0	0	0	3	3	3	3	3	2	2	2	2	2	1
*	*	*	*		*	*	*	*	*	*			*	*			*	*	*

Ghi chú : * : có lỗi trang

LRU

7	**0**	**1**	**2**	**0**	**3**	**0**	**4**	**2**	**3**	**0**	**3**	**2**	**1**	**2**	**0**	**1**	7	**0**	**1**
7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
		1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7
*	*	*	*		*		*	*	*	*			*		*		*		

BÀI TẬP

- Cho chuỗi tham chiếu 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5
- Nếu bộ nhớ có 3 frame. Sử dụng thuật toán FIFO thì có mấy lỗi trang
- Nếu bộ nhớ có 4 frame. Sử dụng thuật toán FIFO thì có mấy lỗi trang
- *Tìm hiểu về "nghịch lý Belady"*

BÀI TẬP

- Giả sử bộ nhớ chính được phân thành các phân vùng có kích thước là 100KB, 500K, 200K, 300K và 600K (theo thứ tự), cho biết các tiến trình có kích thước 212K, 417K, 112K và 426K (theo thứ tự) sẽ được cấp phát bộ nhớ như thế nào, nếu sử dụng :
  a) Thuật toán First fit
  b) Thuật toán Best fit
  c) Thuật toán Worst fit