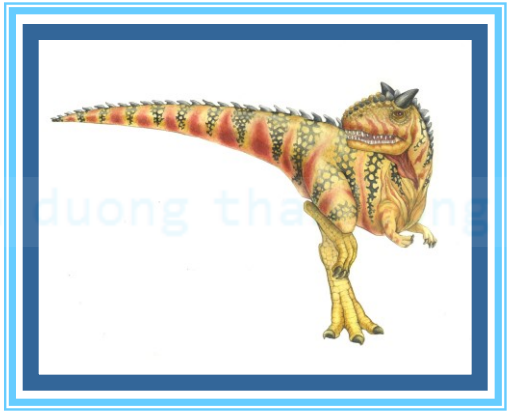
Chương 7: Quản lý bộ nhớ - 2



UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập chương 7-1

- Chuyển đổi địa chỉ là gì? Địa chỉ nhớ được biểu diễn như thế nào trong quá trình chạy một chương trình?
- Khi nào địa chỉ lệnh và dữ liệu được chuyển thành địa chỉ thật?
- Thế nào là dynamic linking? Nêu ưu điểm?
- Thế nào là dynamic loading?
- Nêu cơ chế overlay? Swapping?
- Nêu các mô hình quản lý bộ nhớ?

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập chương 7-1

- Thế nào là phân mảnh ngoại? Phân mảnh nội? Cho ví dụ?
- Fixed partitioning là gì? Các chiến lược placement?
- Dynamic partitioning là gì? Các chiến lược placement?

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Câu hỏi ôn tập chương 7-1

Giả sử bộ nhớ chính được cấp phát các phân vùng có kích thước là 600K, 500K, 200K, 300K (theo thứ tự), sau khi thực thi xong, các tiến trình có kích thước 212K, 417K, 112K, 426K (theo thứ tự) sẽ được cấp phát bộ nhớ như thế nào, nếu sử dụng: Thuật toán First fit, Best fit, Next fit, Worst fit? Thuật toán nào cho phép sử dụng bộ nhớ hiệu quả nhất trong trường hợp trên

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Mục tiêu

- Hiểu và vận dụng các cơ chế quản lý bộ nhớ:
  - Cơ chế phân trang
  - Cơ chế phân đoạn

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Nội dung

- Cấp phát không liên tục
  - Cơ chế phân trang
  - Cơ chế phân đoạn
  - Cơ chế kết hợp phân trang và phân đoạn

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Cấp phát không liên tục

- Cơ chế phân trang
- Cơ chế phân đoạn
- Cơ chế kết hợp giữa phân trang và phân đoạn

cuu duong than cong . com

Cơ chế phân trang

- **Bộ nhớ vật lý** thật (của một **hệ thống máy tính**) được chia thành nhiều khối kích thước bằng nhau, gọi là khung trang (**frame**)
- **Bộ nhớ luận lý** (của một **process**) cũng được chia thành nhiều khối kích thước bằng nhau (và cũng bằng kích thước của frame trong bộ nhớ vật lý), gọi là trang (**page**)
- Các chú ý:
  - Kích thước/dung lượng (size) của frame hay page là lũy thừa của 2 (Thường từ khoảng 512 byte đến 16 MB. Một số hệ thống, kích thước 1 trang có thể lên đến 1GB)
  - Các hệ thống hiện nay, địa chỉ vật lý và luận lý hoàn toàn tách biệt nhau. Ví dụ một process có thể có không gian địa chỉ 64-bit (tức dùng 64 bit để định một địa chỉ → bộ nhớ luận lý tương ứng này có tới 2^{64} byte/word) mặc dù bộ nhớ vật lý thật có ít hơn 2^{64} byte/word

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Cơ chế phân trang

- Để quản lý các **page** (biết page nào khi đưa vào bộ nhớ vật lý sẽ được nạp vào frame nào tương ứng), **process** dùng **page table** (Bảng phân trang)

  Bảng phân trang (page table) dùng hỗ trợ ánh xạ địa chỉ luận lý thành địa chỉ vật lý (địa chỉ thực)

- Để quản lý các **frame** (biết frame nào còn trống, frame nào không …), **hệ điều hành** dùng **frame table**

Cơ chế phân trang (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

0	0
1	1
2	2
3	3

Process A page table

0	—
1	—
2	—

Process B page table

0	7
1	8
2	9
3	10

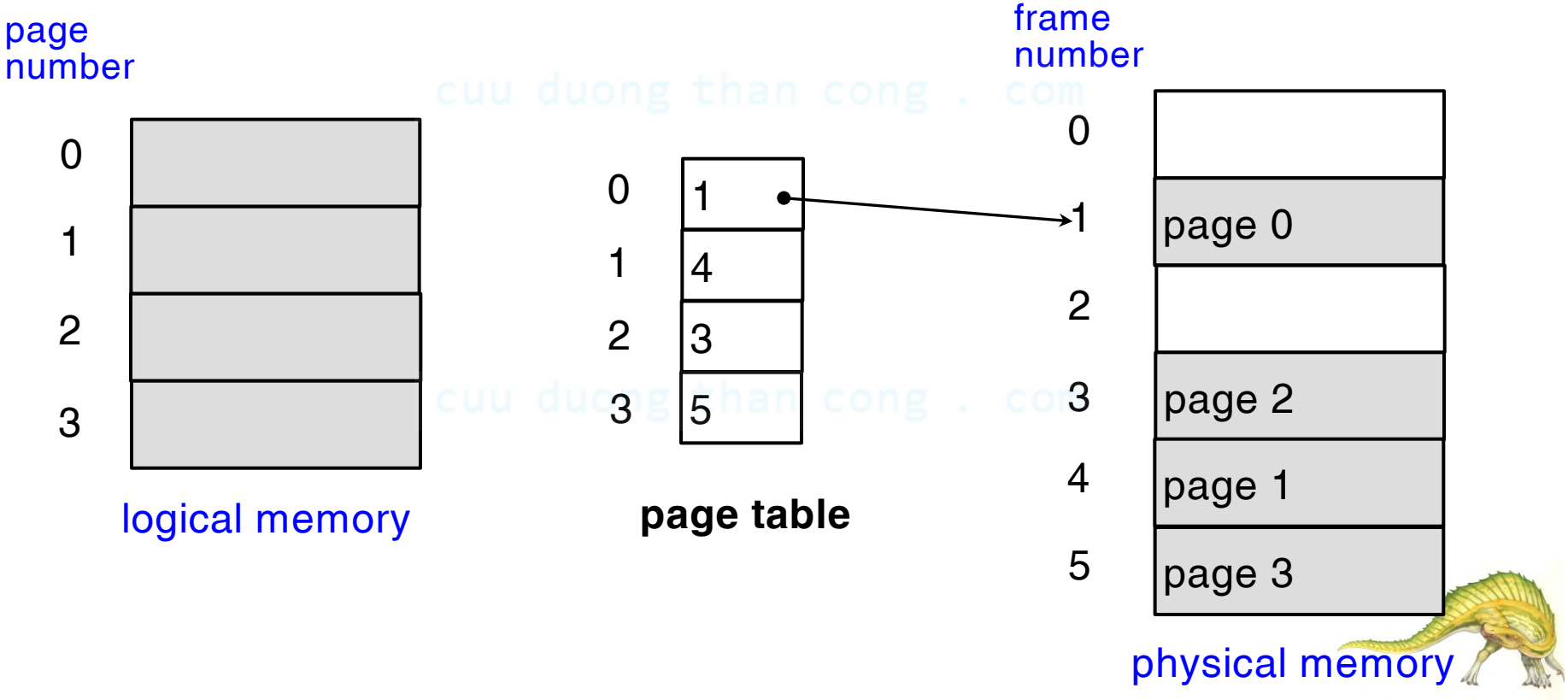
Process C page table

0	4
1	5
2	6
3	11
4	12

Process D page table

13
14

Free frame list



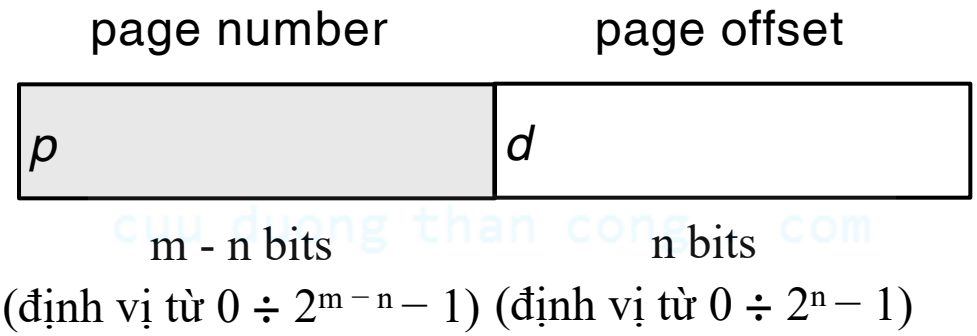
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Cơ chế phân trang (tt)

- Chuyển đổi địa chỉ trong paging
- Cài đặt bảng trang
- Effective access time
- Tổ chức bảng trang
- Bảo vệ bộ nhớ

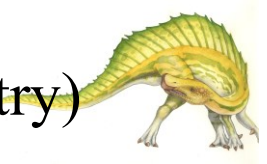
Chuyển đổi địa chỉ trong paging

- Địa chỉ luận lý gồm có:
  - Số hiệu trang (Page number) p
  - Địa chỉ tương đối trong trang (Page offset) d
- Nếu kích thước của không gian địa chỉ ảo là 2^m byte/word, và kích thước của mỗi trang là 2^n byte/word (đơn vị là byte hay word tùy theo kiến trúc máy) thì

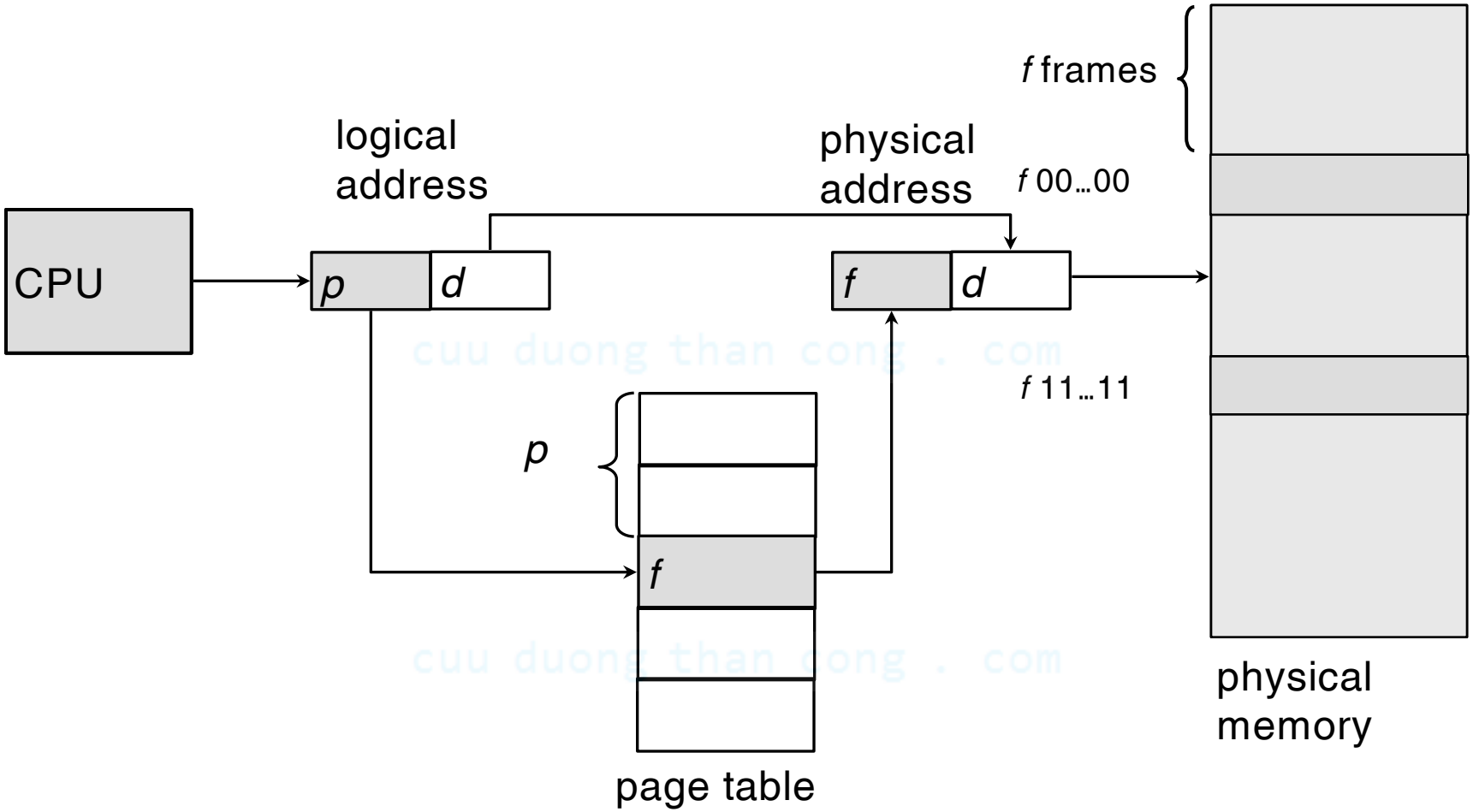


➔ Có tổng cộng $2^m/2^n = 2^{m-n}$ trang

➔ Bảng phân trang (Page table) sẽ có tổng cộng 2^{m-n} mục (entry)



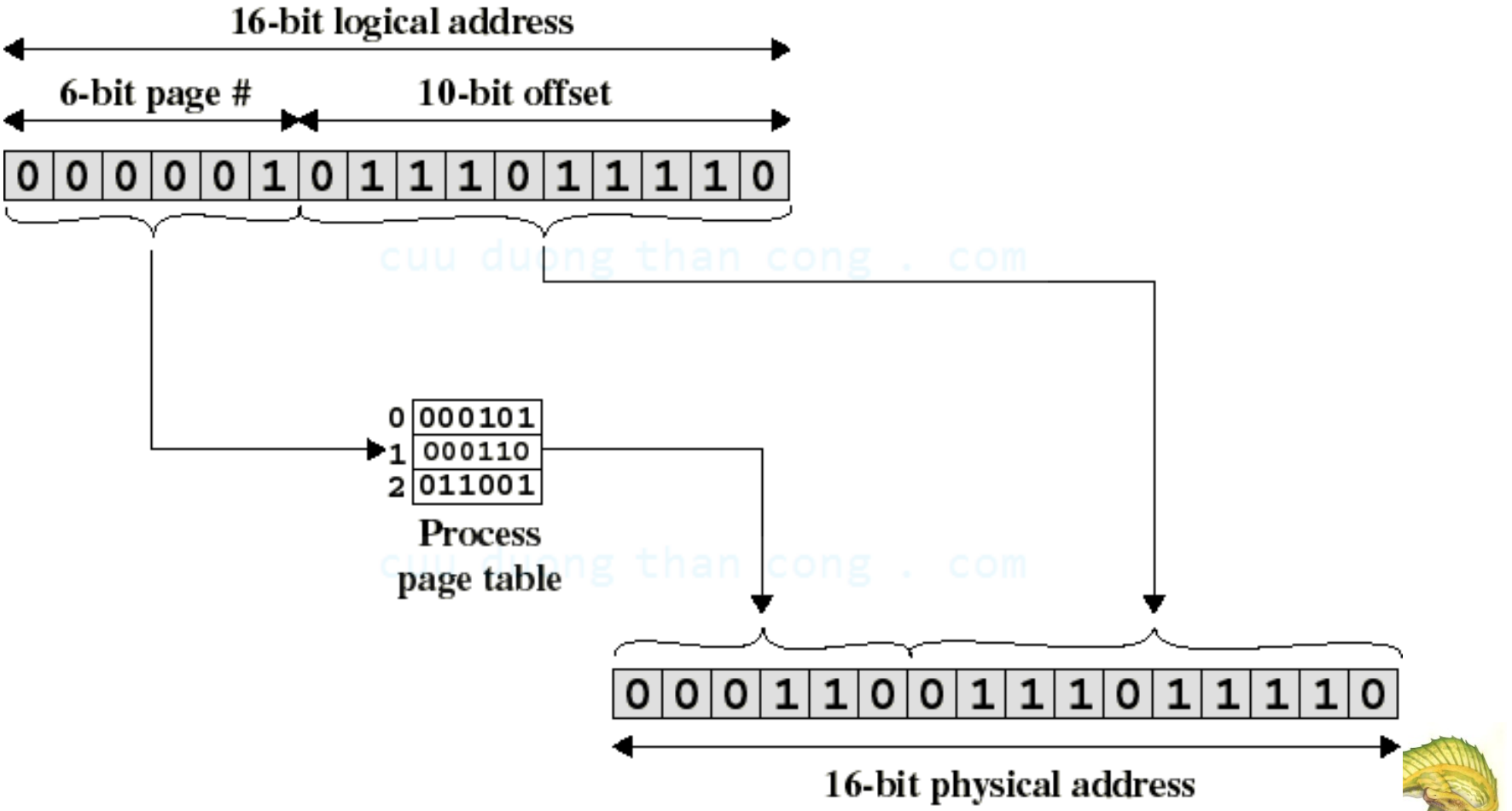
Chuyển đổi địa chỉ trong paging (tt)

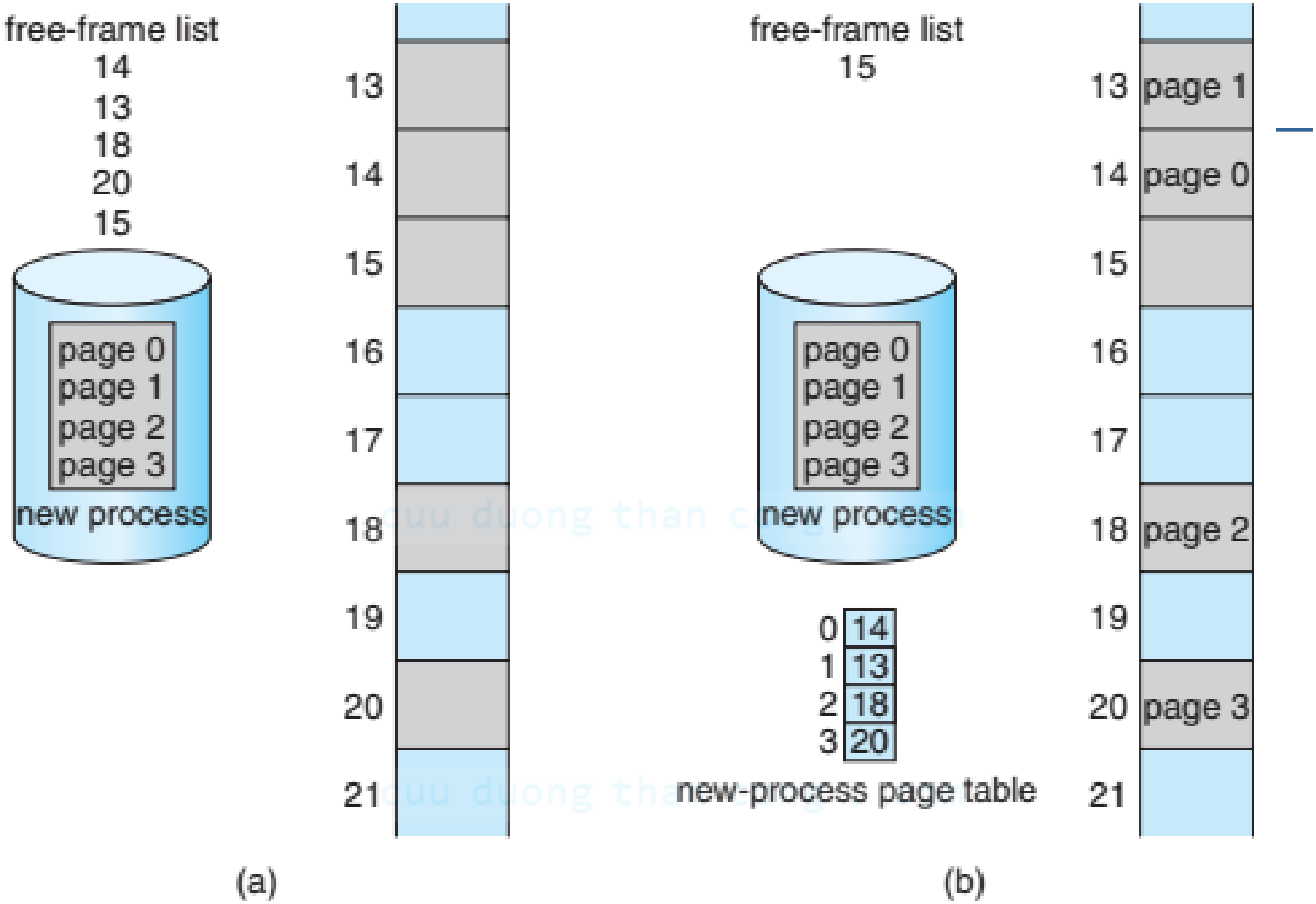


Chuyển đổi địa chỉ trong paging (tt)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Ví dụ:





(a): Trước khi các page trong một new process được cấp phát vào các frame trống trong bộ nhớ vật lý.

(b): Sau khi các page trong một new process được cấp phát vào các frame trống trong bộ nhớ vật lý.

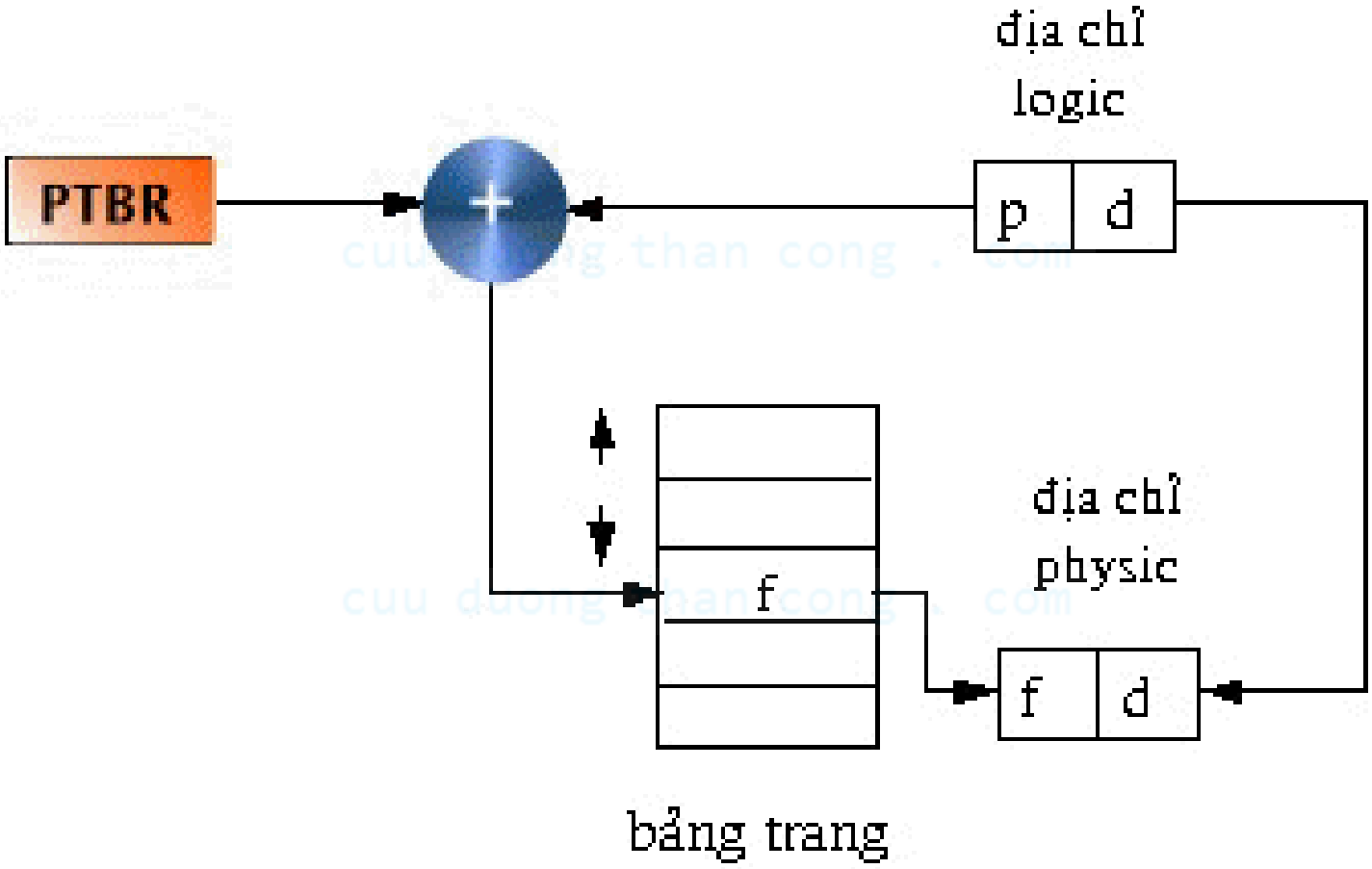
Cài đặt bảng trang (paging hardware)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Bảng phân trang thường được lưu giữ trong bộ nhớ chính
  - Mỗi process được hệ điều hành cấp một bảng phân trang
  - Thanh ghi page-table base (PTBR) trỏ đến bảng phân trang
  - Thanh ghi page-table length (PTLR) biểu thị kích thước của bảng phân trang (có thể được dùng trong cơ chế bảo vệ bộ nhớ)
- Thường dùng một bộ phận cache phần cứng có tốc độ truy xuất và tìm kiếm cao, gọi là thanh ghi kết hợp (associative register) hoặc translation look-aside buffers (TLBs)

Cài đặt bảng trang (tt)

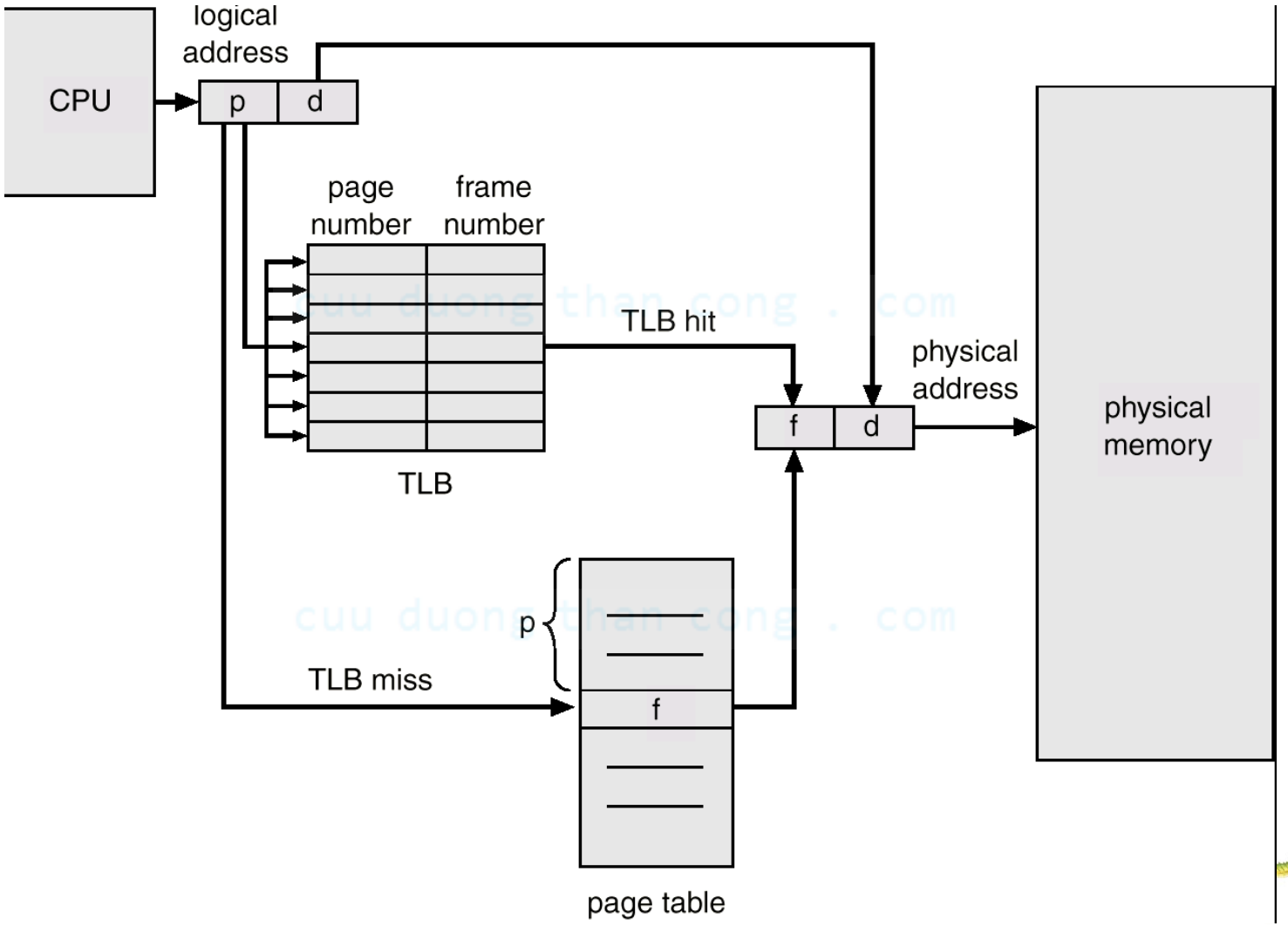
- Cách dùng thanh ghi Page-Table Base Register (PTBR)



Cài đặt bảng trang (tt)

UIT
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

Cài đặt bảng trang có cải tiến với TLB (**Translation look-aside buffer**)



Effective access time (EAT)

Tính thời gian truy xuất hiệu dụng (effective access time, EAT):

- Thời gian tìm kiếm trong TLB (associative lookup): ε
- Thời gian một chu kỳ truy xuất bộ nhớ: x
- Hit ratio: tỉ số giữa số lần chỉ số trang được tìm thấy (hit) trong TLB và số lần truy xuất khởi nguồn từ CPU.

  Kí hiệu hit ratio: α

Thời gian cần thiết để có được chỉ số frame:

- Khi chỉ số trang có trong TLB (hit): ε + x
- Khi chỉ số trang không có trong TLB (miss): ε + x + x

➔ **Thời gian truy xuất hiệu dụng:**

$$EAT = (\varepsilon + x)\alpha + (\varepsilon + 2x)(1 - \alpha)$$

$$= (2 - \alpha)x + \varepsilon$$

Effective access time (EAT) (tt)

- Ví dụ 1: đơn vị thời gian nano giây
  - Associative lookup = 20
  - Memory access = 100
  - Hit ratio = 0.8
  - EAT = (100 + 20) × 0.8 + (200 + 20) × 0.2
    = 1.2 × 100 + 20
    = 140
- Ví dụ 2: đơn vị thời gian nano giây
  - Associative lookup = 20
  - Memory access = 100
  - Hit ratio = 0.98
  - EAT = (100 + 20) × 0.98 + (200 + 20) × 0.02
    = 1.02 × 100 + 20
    = 122

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Tổ chức bảng trang

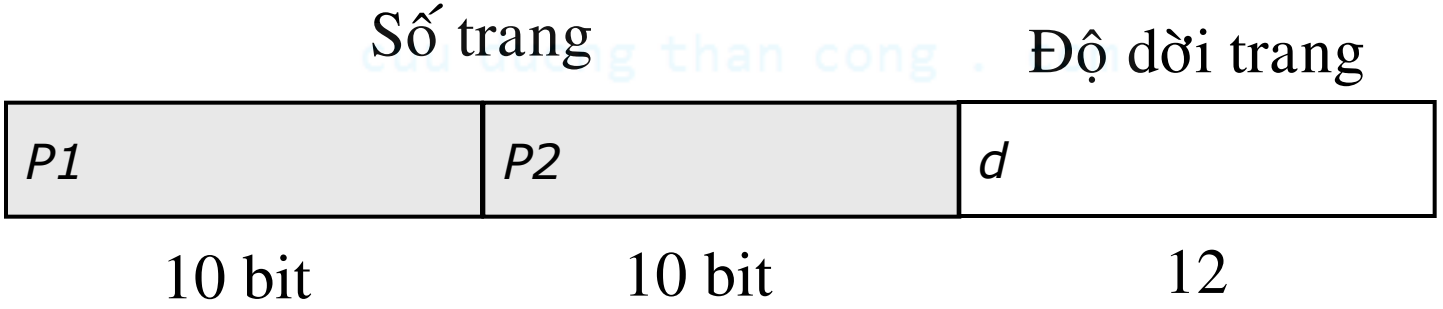
Bảng phân trang thường được tổ chức/cấu trúc theo 3 kiểu:

- Cấu trúc bảng phân trang theo kiểu kế thừa (Phân trang 2 cấp) - Hierarchical paging
- Cấu trúc bảng phân trang theo kiểu nghịch đảo – Inverted page tables
- Cấu trúc bảng phân trang theo kiểu dùng hash function - Hashed page tables

Slide sau trình chỉ bày 2 kiểu: bảng phân trang 2 cấp và nghịch đảo

Tổ chức bảng trang 2 cấp

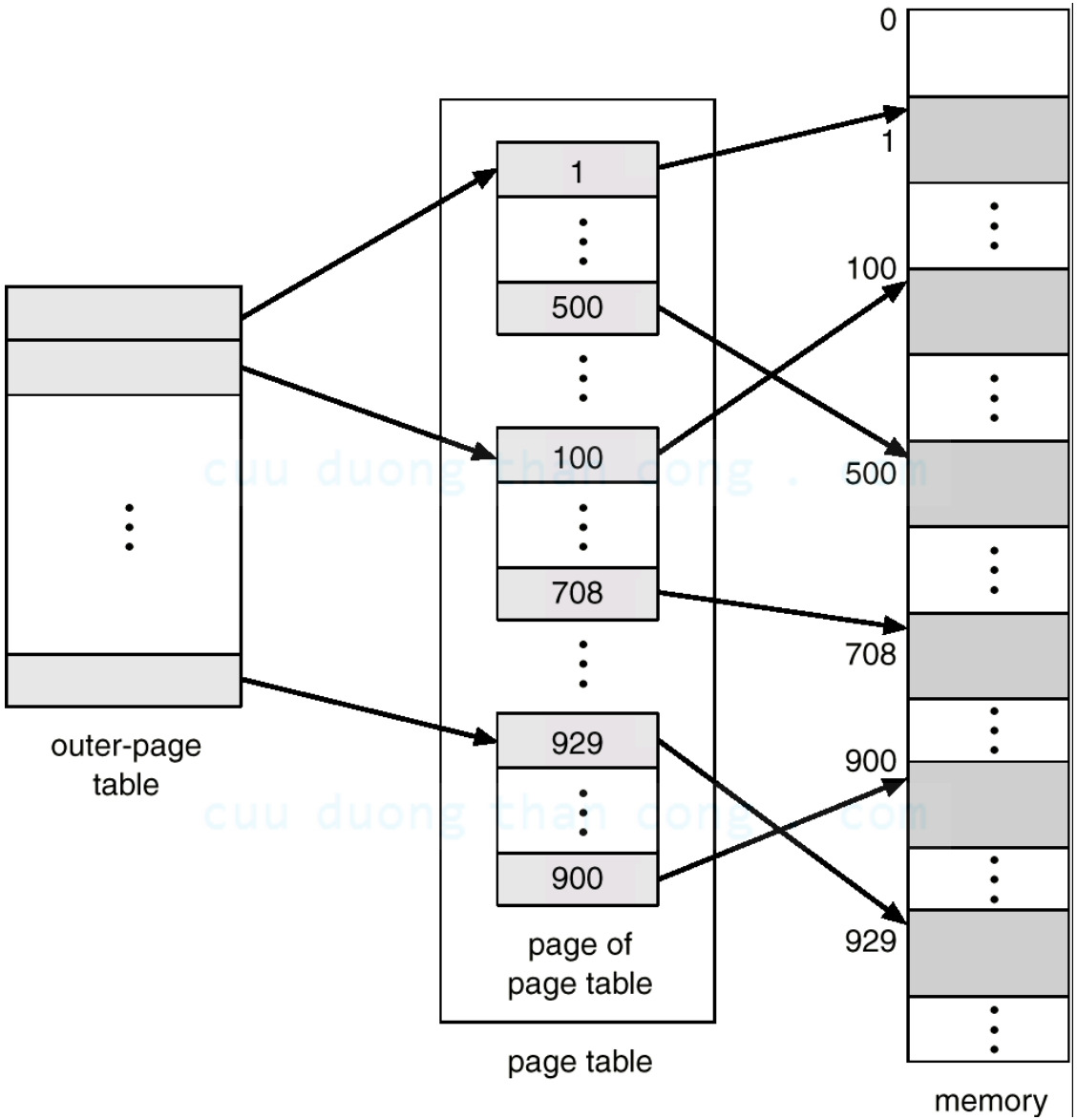
- Các hệ thống hiện đại đều hỗ trợ không gian địa chỉ ảo rất lớn (2^{32} đến 2^{64}), ở đây giả sử là 2^{32}
  - Giả sử kích thước trang nhớ là 4KB (= 2^{12}) ⇒ bảng phân trang sẽ có $2^{32}/2^{12} = 2^{20}$ = 1M mục.
  - Giả sử mỗi mục gồm 4 byte thì mỗi process cần 4MB cho bảng phân trang
  - Vì 4MB là khá nhiều, để tiện việc tìm kiếm trong bảng phân trang, lúc này bản phân trang cũng được phân trang trong nó → Phân trang 2 cấp



Ví dụ với phân trang 2 cấp

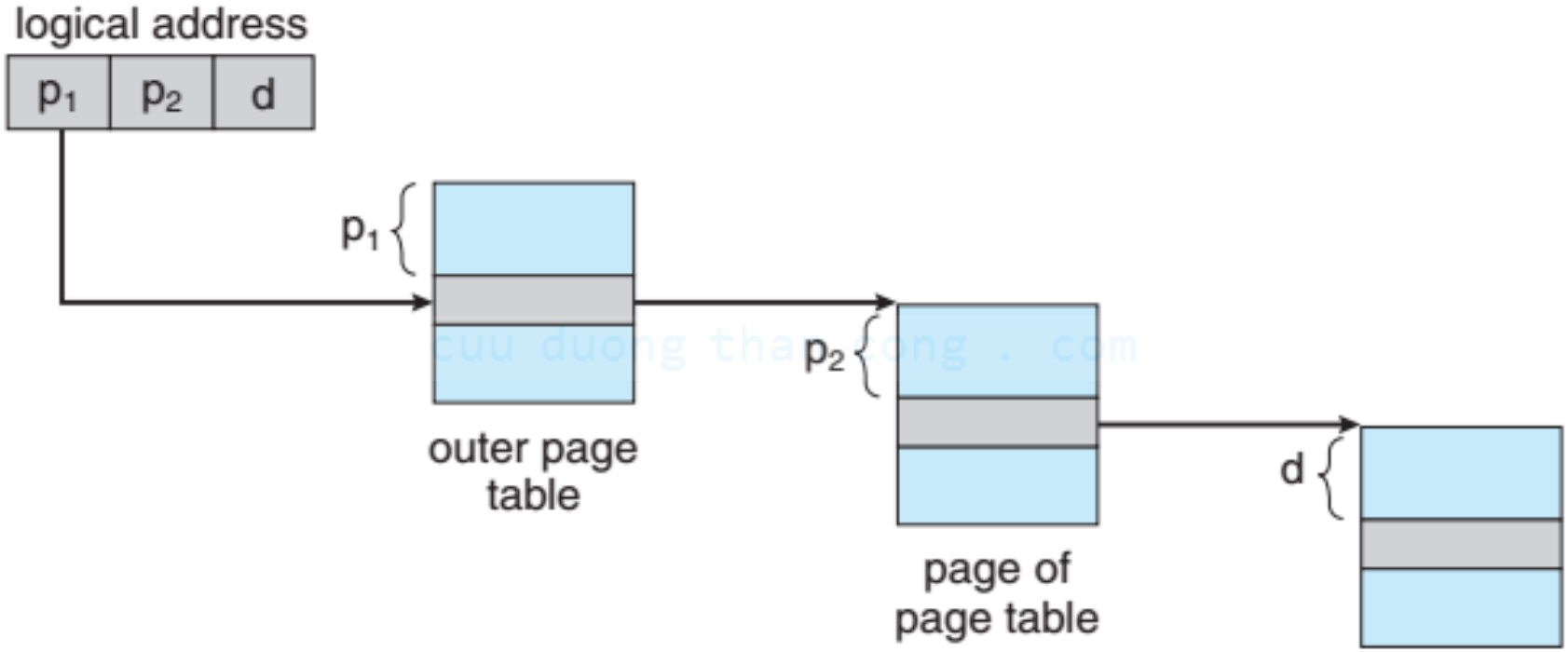
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Tổ chức bảng trang 2 cấp



UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

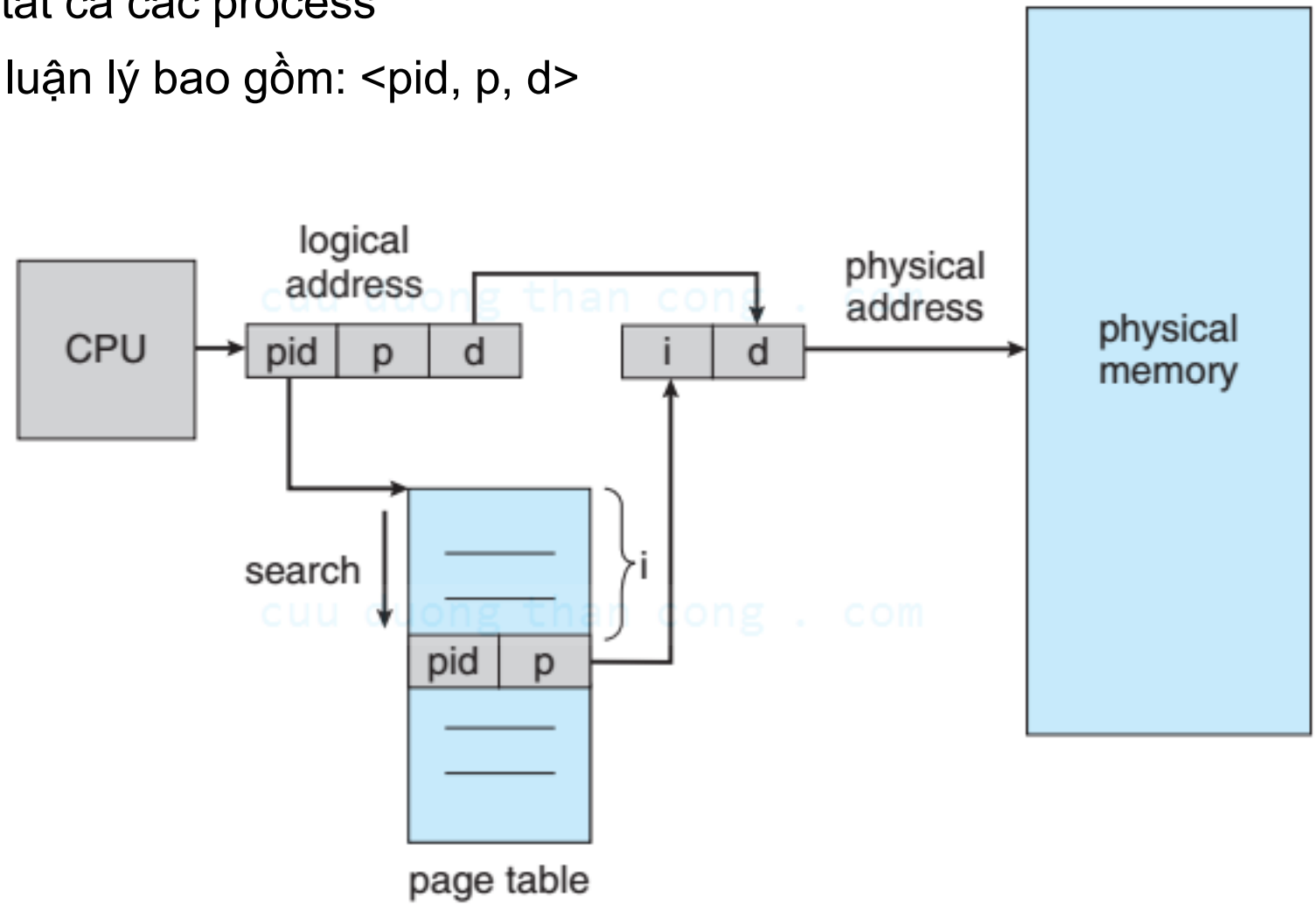
Tổ chức bảng trang 2 cấp



Tổ chức bảng trang nghịch đảo

- Bảng trang nghịch đảo (IBM system/38, IBM RISC, IBM RT): sử dụng cho tất cả các process

Địa chỉ luận lý bao gồm: <pid, p, d>

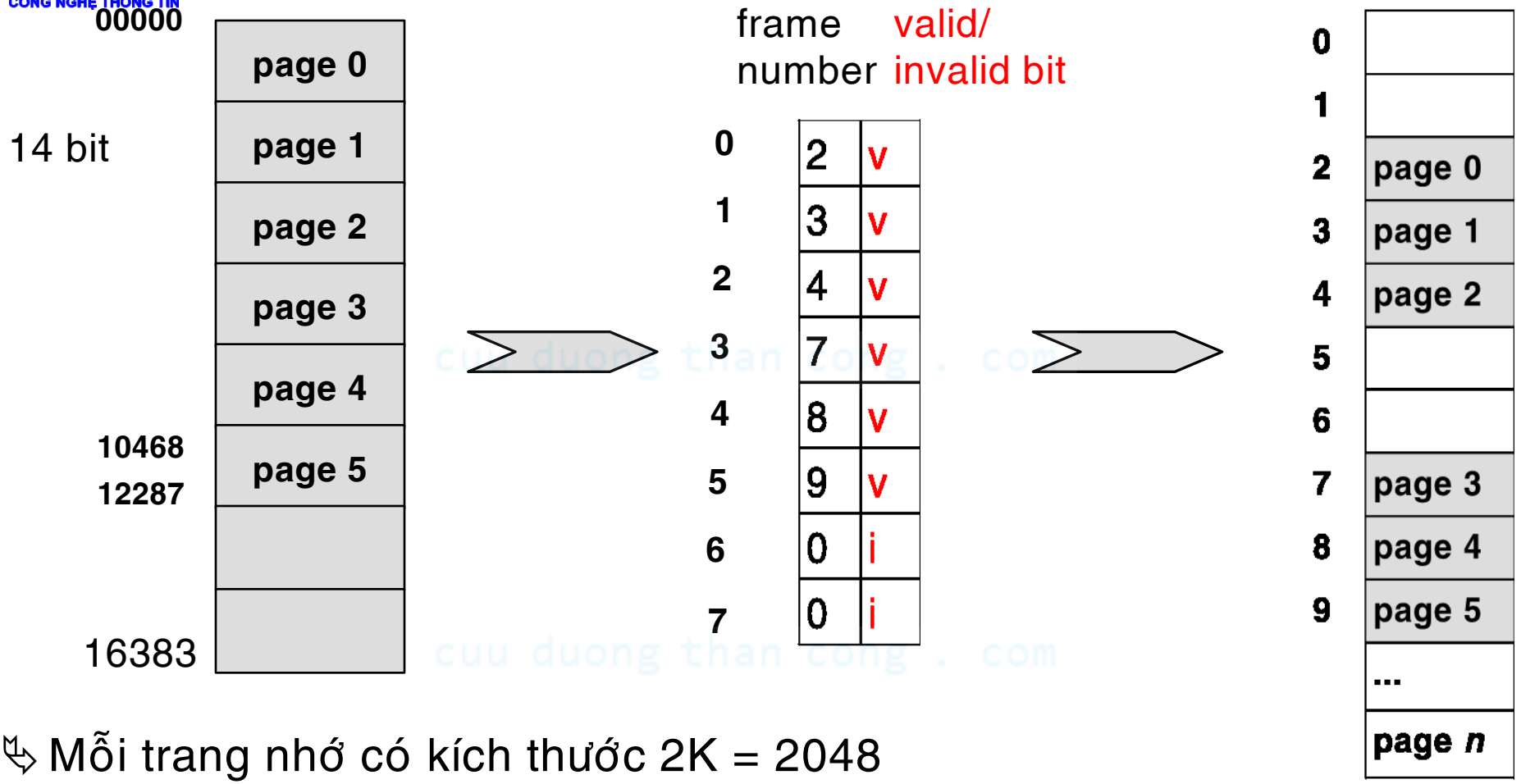


UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Bảo vệ bộ nhớ

- Việc bảo vệ bộ nhớ có thể được hiện thực bằng cách: các bit bảo vệ frame được giữ kèm trong bảng phân trang. Các bit này biểu thị các thuộc tính sau:
  - read-only, read-write, execute-only
- Ngoài ra, còn có một valid/invalid bit gắn với mỗi mục trong bảng phân trang
  - "valid": cho biết là trang của process, do đó là một trang hợp lệ.
  - "invalid": cho biết là không là trang của process, do đó là một trang bất hợp lệ.

Bảo vệ bằng valid/invalid bit



- Mỗi trang nhớ có kích thước 2K = 2048
- Process có kích thước 10,468 ⇒ phân mảnh nội ở frame 9 (chứa page 5), các địa chỉ ảo > 12287 là các địa chỉ invalid.

Chia sẻ các trang nhớ

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Process 1

ed 1	
ed 2	
ed 3	
data 1	

0	3
1	4
2	6
3	1

Process 2

ed 1	
ed 2	
ed 3	
data 2	

0	3
1	4
2	6
3	7

ed 1	
ed 2	
ed 2	
data 3	

0	3
1	4
2	6
3	2

Process 3

0	
1	data 1
2	data 3
3	ed 1
4	ed 2
5	
6	ed 3
7	data 2
8	
9	
10	

Bợ nhớ thực

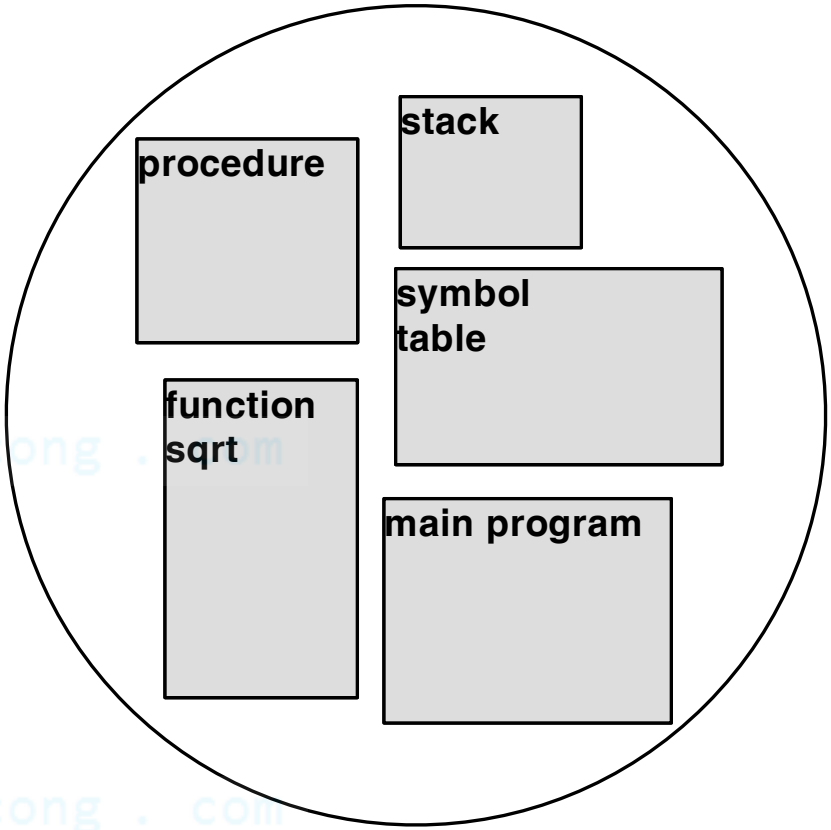
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Phân đoạn (segmentation)

- Nhìn lại cơ chế phân trang
  - User view (không gian địa chỉ ảo) tách biệt với không gian bộ nhớ thực. Cơ chế phân trang thực hiện phép ánh xạ user-view vào bộ nhớ thực.
- Trong thực tế, dưới góc nhìn của user, một chương trình cấu thành từ nhiều đoạn (segment). Mỗi đoạn là một đơn vị luận lý của chương trình, như
  - main program, procedure, function
  - local variables, global variables, common block, stack, symbol table, arrays,…

User view của một chương trình

- Thông thường, một chương trình được biên dịch. Trình biên dịch sẽ tự động xây dựng các segment.
- Ví dụ, trình biên dịch tạo ra các segment như hình bên:
  - Stack
  - Symbol table
  - Procedure/function code
- Trình loader sẽ gán mỗi segment một số định danh riêng.



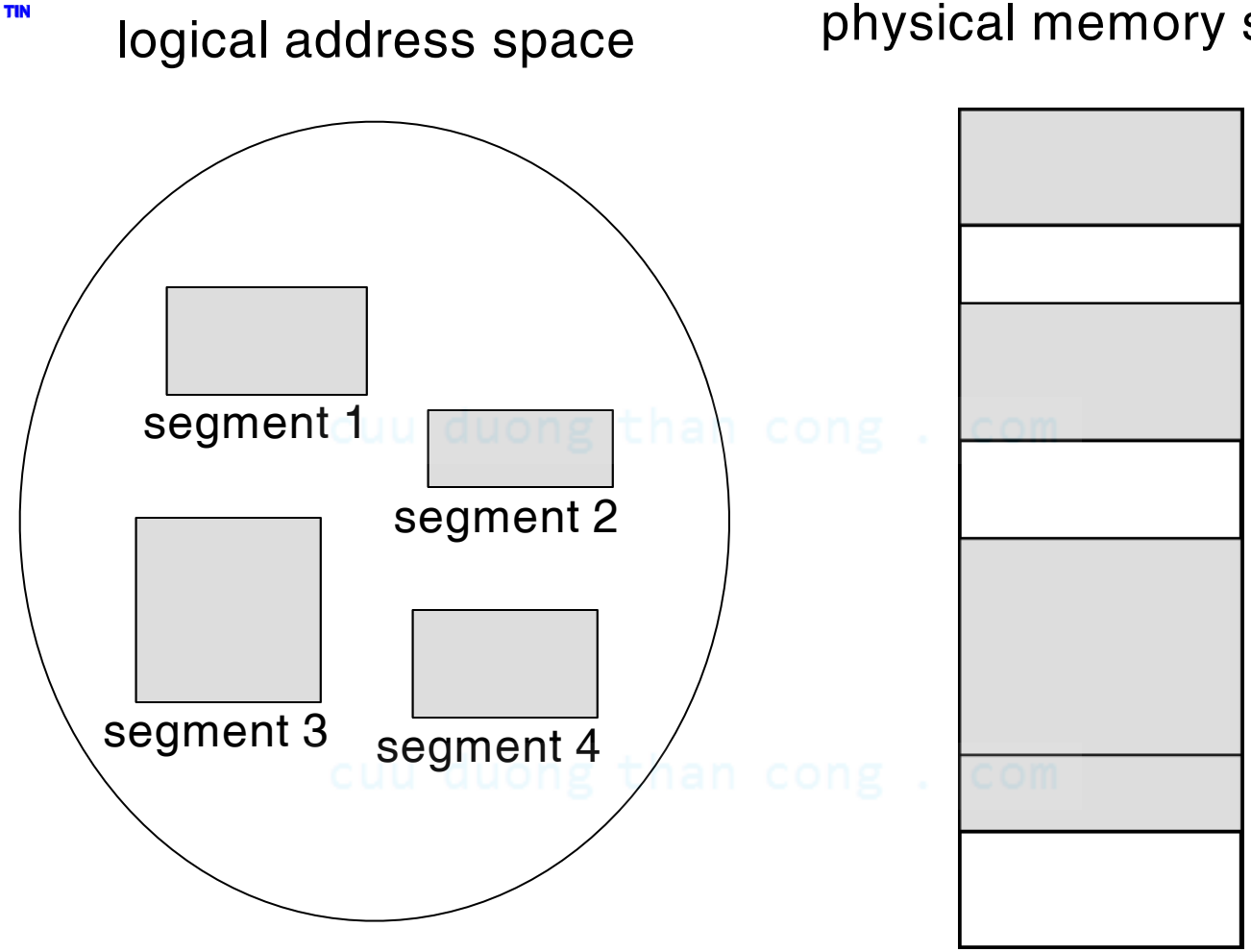
Logical address space

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Phân đoạn

- Dùng cơ chế phân đoạn để quản lý bộ nhớ có hỗ trợ user view
  - Không gian địa chỉ ảo là một tập các đoạn, mỗi đoạn có tên và kích thước riêng.
  - Một địa chỉ luận lý được định vị bằng tên đoạn và độ dời (offset) bên trong đoạn đó (so sánh với phân trang!)

Phân đoạn (tt)


logical address space
physical memory space
segment 1
segment 2
segment 3
segment 4

Cài đặt phân đoạn

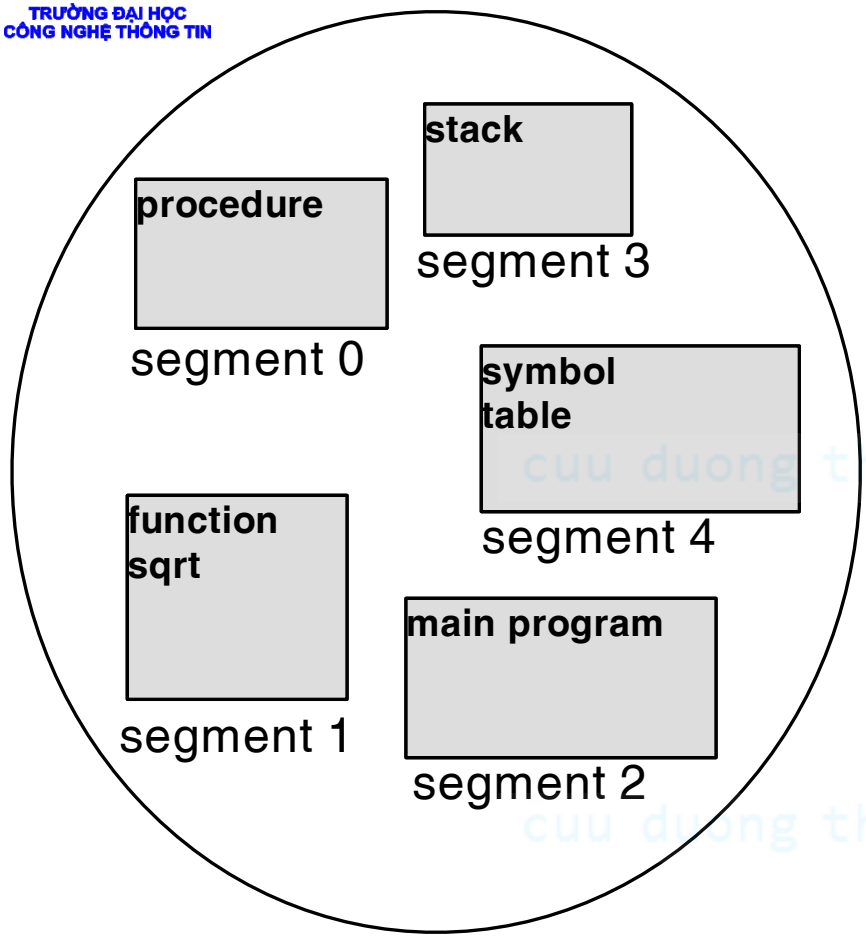
- Địa chỉ luận lý là một cặp giá trị

  <segment number, offset>

- Bảng phân đoạn (**segment table**): gồm nhiều mục (item), mỗi mục gồm **limit** và **base** với:
  - base: chứa địa chỉ khởi đầu của segment trong bộ nhớ
  - limit: xác định kích thước của segment
- **Segment-table base register (STBR)**: trỏ đến vị trí bảng phân đoạn trong bộ nhớ
- **Segment-table length register (STLR)**: số lượng segment của chương trình

⇒ Một chỉ số segment s là hợp lệ nếu s < STLR

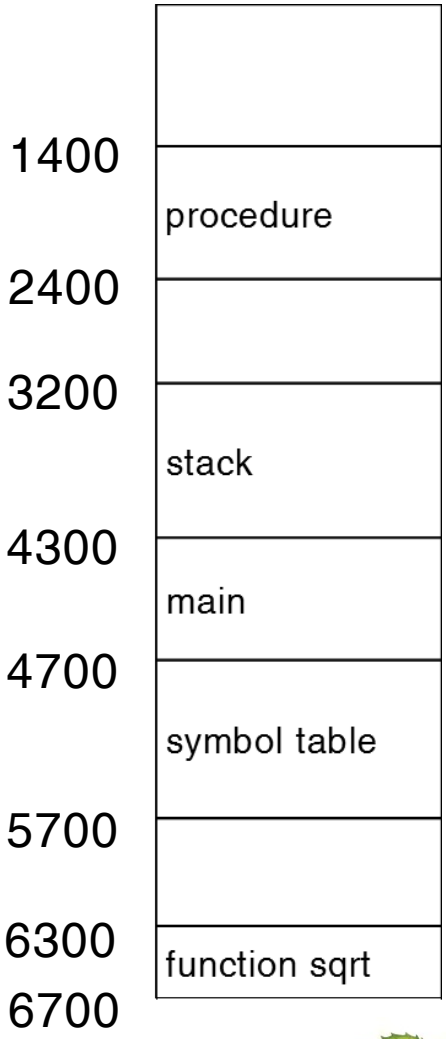
Một ví dụ về phân đoạn



logical address space

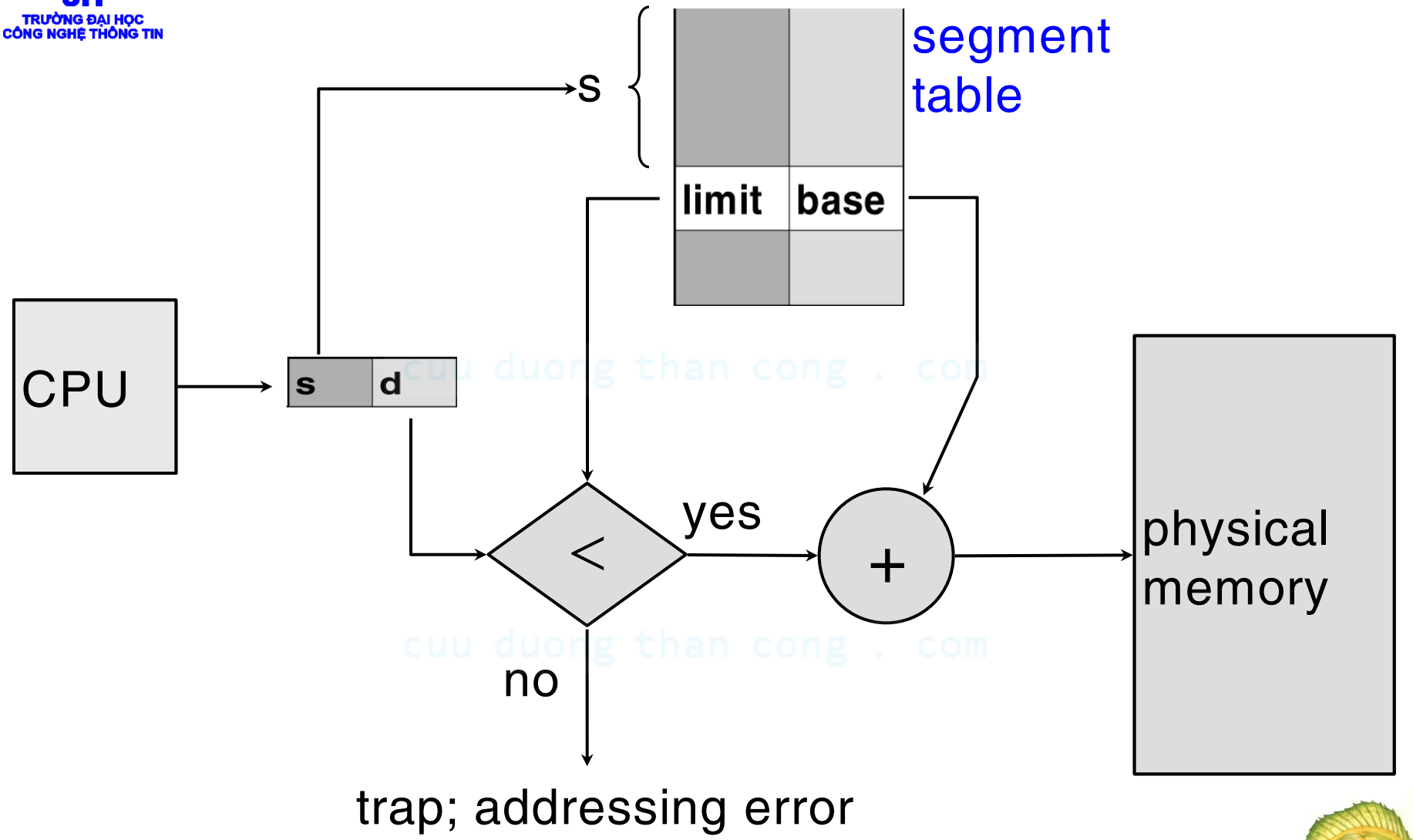
	limit	base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

segment table



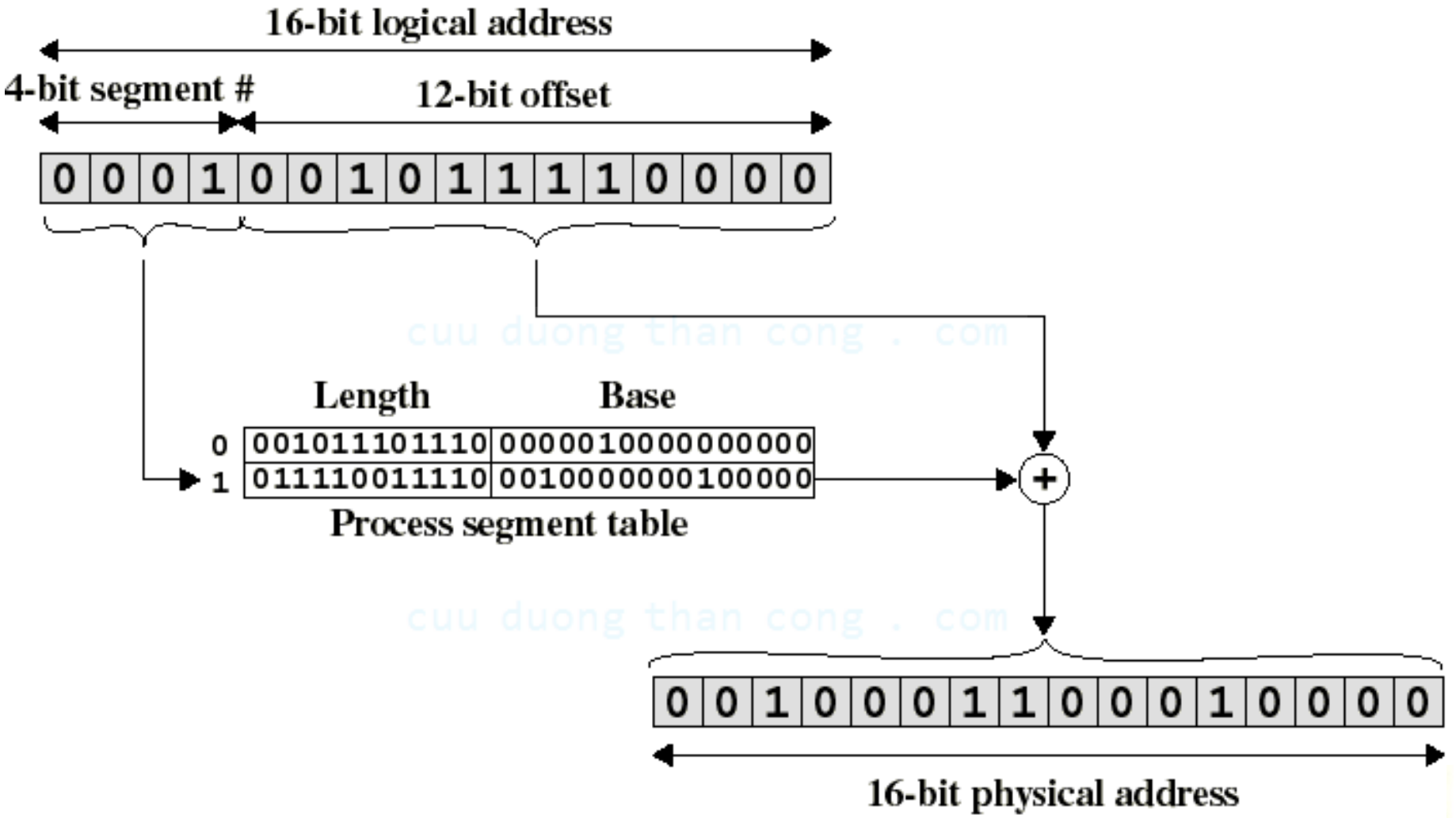
physical memory space

Phần cứng hỗ trợ phân đoạn

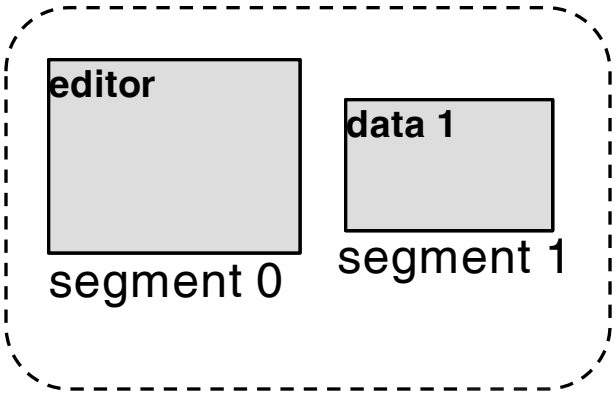


Chuyển đổi địa chỉ trong cơ chế phân đoạn

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN



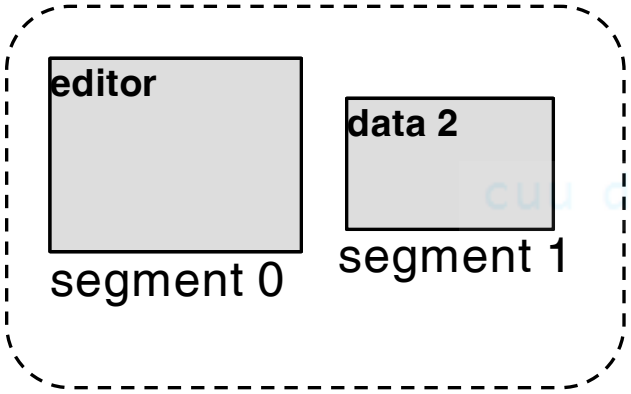
Chia sẻ các đoạn



logical address space
process P1

	limit	base
0	25286	43062
1	4425	68348

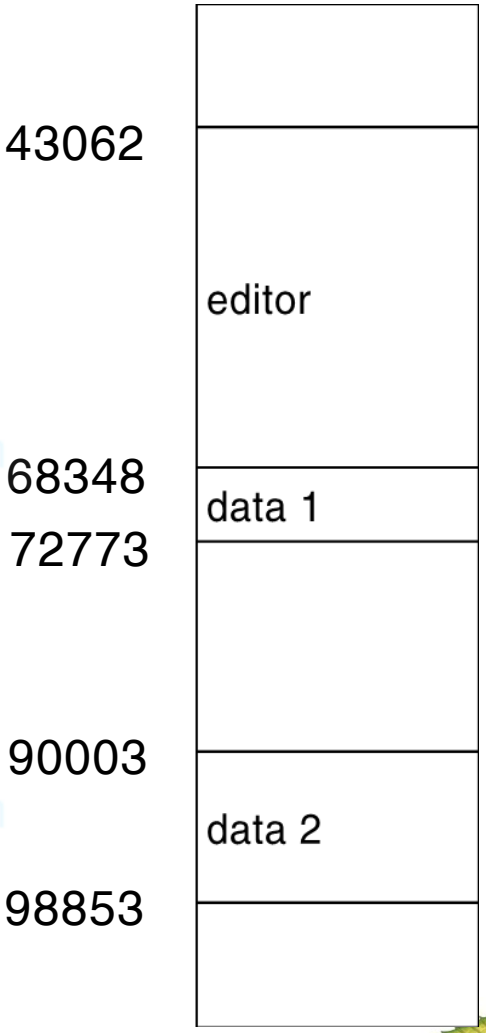
segment table
process P_1



logical address space
process P2

	limit	base
0	25286	43062
1	8850	90003

segment table
process P_2



physical memory

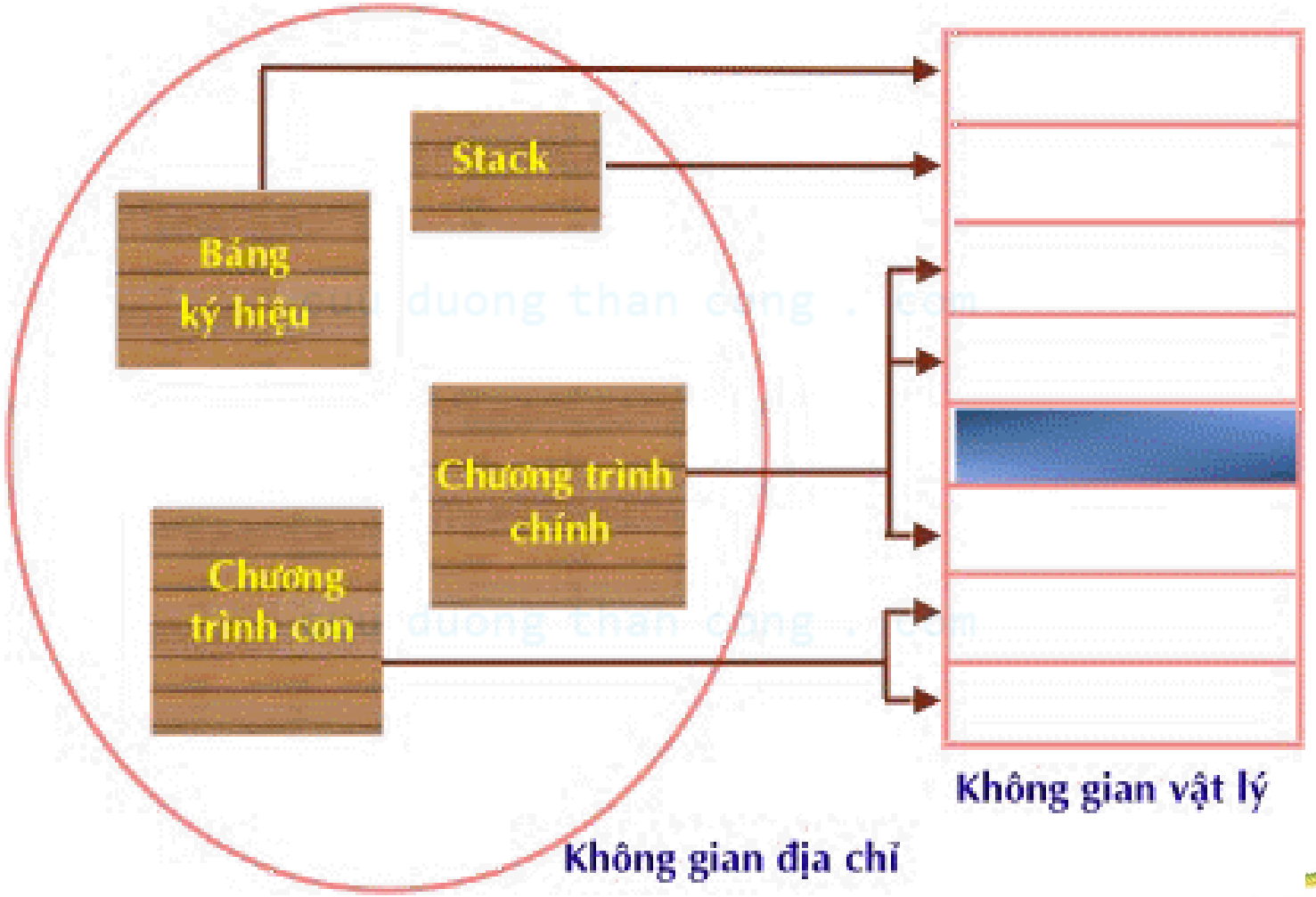
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Kết hợp phân trang và phân đoạn

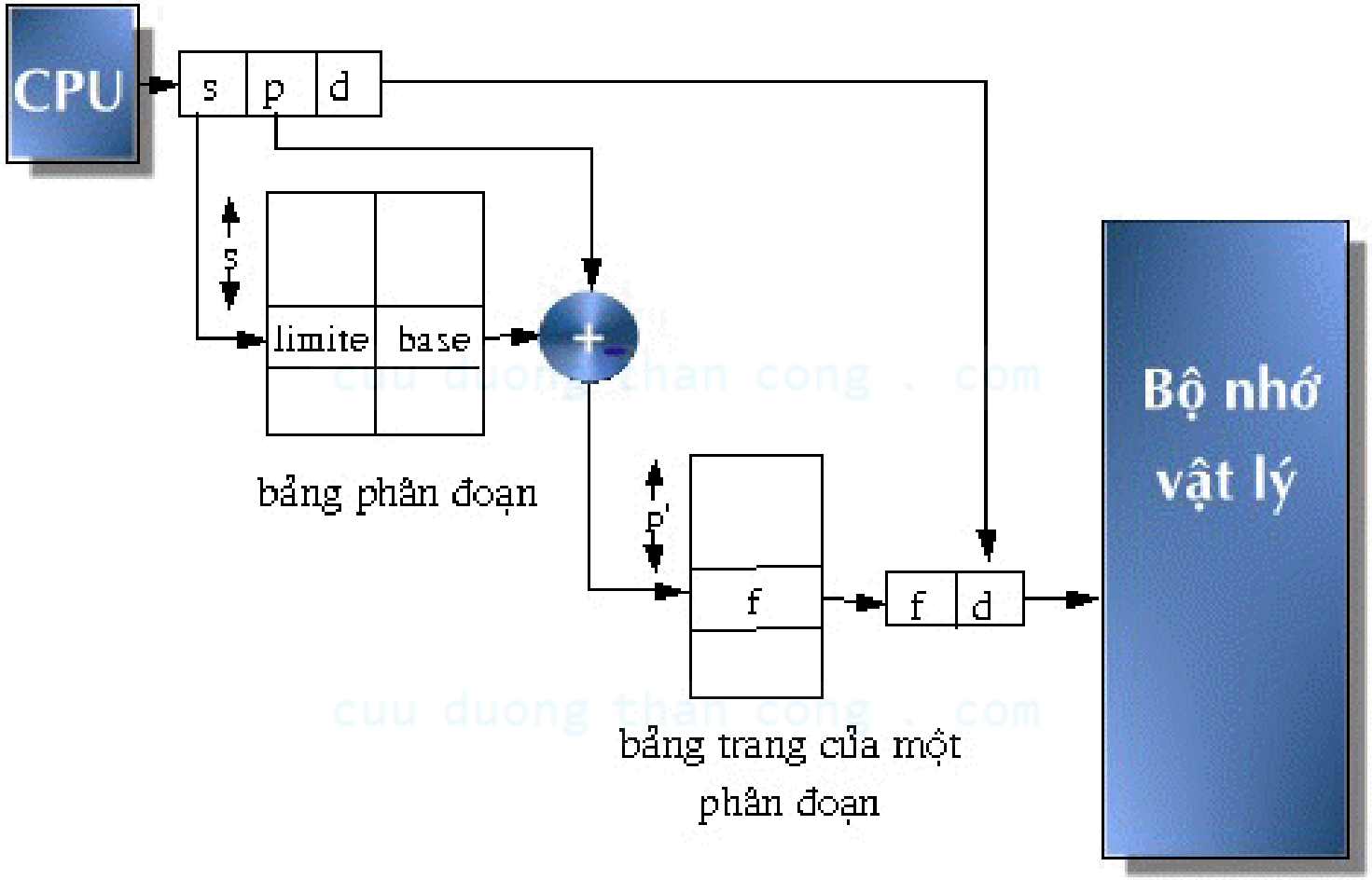
- Kết hợp phân trang và phân đoạn nhằm kết hợp các ưu điểm đồng thời hạn chế các khuyết điểm của phân trang và phân đoạn:
  - Vấn đề của phân đoạn: Nếu một đoạn quá lớn thì có thể không nạp nó được vào bộ nhớ.
  - Ý tưởng giải quyết: paging đoạn, khi đó chỉ cần giữ trong bộ nhớ các page của đoạn hiện đang cần.

Logic Addr = <s, p, d>

Kết hợp phân trang và phân đoạn (tt)



Cài đặt phân đoạn



Hình 4.23 Cơ chế phần cứng của sự phân đoạn kết hợp phân trang

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Ôn tập

- Cấp phát không liên tục
  - Cơ chế phân trang
  - Cơ chế phân đoạn
  - Cơ chế kết hợp phân trang và phân đoạn

Kết thúc chương 7