**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**



**KHOA CÔNG NGHỆ THÔNG TIN**

****

**BÀI TẬP LỚN**

**Môn: Nguyên lý hệ điều hành**

* **ĐỀ TÀI -**

**Nghiên cứu tìm hiều về quản lí bộ nhớ trong**

**Hệ Điều Hành Windows**

**Giảng viên hướng dẫn: Nguyễn Thanh Hải.**

**Lớp 20212IT6025008 Khóa 15**

**Nhóm sinh viên thực hiện : Nhóm 3**

**Thành viên:**

**1. Trần Hải Nam**

**2. Trần Tuấn Hùng**

**3. Nguyễn Quang Trường**

**4. Nguyễn Hoàng Giang**

**5. Nguyễn Hoài Nam**

**Hà Nội, tháng 5 năm 2022**

MỤC LỤC

[I.Dẫn nhập và khái niệm : 1](#_Toc105170505)

[1. Dẫn nhập: 1](#_Toc105170506)

[2. Khái niệm: 1](#_Toc105170507)

[II.Hệ thống Windows quản lí bộ logic theo cấu trúc phân trang (paging) : 1](#_Toc105170508)

[1. Physical Storage : 1](#_Toc105170509)

[2. Virtual Address Space (Không gian địa chỉ ảo): 2](#_Toc105170510)

[3. Phân trang ( paging ): 5](#_Toc105170511)

[4. Windows Page Table Management (Quản lý bảng trang Windows): 8](#_Toc105170512)

[5. Windows Memory Protection (Bảo vệ bộ nhớ Windows): 9](#_Toc105170513)

[III. Quản lí bộ nhớ ảo ( bộ nhớ Logic ) : 10](#_Toc105170514)

[1. Tổ chức bộ nhớ ảo 10](#_Toc105170515)

[2. Không gian địa chỉ ảo (virtual address space) 11](#_Toc105170516)

[3. Phân trang nhu cầu 12](#_Toc105170517)

[4. Page Fault 14](#_Toc105170518)

[5. Bộ đệm giao diện dịch (Translation Lookaside Buffer - TLB) 14](#_Toc105170519)

[6. Phân đoạn 14](#_Toc105170520)

[7. Kết hợp phân đoạn với phân trang 16](#_Toc105170521)

[8. Bảng trang đảo ngược (Inverted Page Table) 17](#_Toc105170522)

[IV. Quản lí bộ nhớ vật lí: 17](#_Toc105170523)

[1. Phân chia vùng trong RAM: 17](#_Toc105170524)

[2. Cách thức chuyển đổi giữa các vùng trong RAM: 18](#_Toc105170525)

[3. CSDL về khung trang: 19](#_Toc105170526)

**Lời Mở Đầu**

Bộ nhớ là tài nguyên quan trọng thứ 2 sau CPU trong một hệ thống máy tính. Bộ nhớ bao gồm các byte hoặc các từ được đánh địa chỉ. Đây là chỗ chứa các tiến trình và dữ liệu của tiến trình. Việc quản lí và sử dụng bộ nhớ hợp lí ảnh hưởng tới tốc độ và khả năng của toàn bộ hệ thống tính toán. Do vậy quản lí bộ nhớ là 1 chức năng quan trọng của HĐH.

Các công việc liên quan tới quản lí bộ nhớ bao gồm quản lí bộ nhớ trống, cấp phát bộ nhớ trống cho các tiến trình và giải phóng bộ nhớ đã cấp phát, ngăn chặn việc truy xuất trái phép tới các vùng bộ nhớ, ánh xạ giữa các địa chỉ logic và địa chỉ vật lí. Trong trường hợp yêu cầu về bộ nhớ của các tiến trình lớn hơn dung lượng bộ nhớ vật lí HĐH cho phép trao đổi thông tin giữa đĩa và bộ nhớ hoặc tổ chức bộ nhớ ảo để thỏa mãn nhu cầu của tiến trình.

Chúng em chân thành cảm ơn sự hướng dẫn, chỉ bảo tận tình của thầy Nguyễn Thanh Hải. Trong quá trình làm bài không tránh khỏi sai sót, chúng em rất mong nhận được những ý kiến đóng góp của thầy và những người quan tâm để được hoàn thiện hơn ạ.

Chúng em xin chân thành cảm ơn!

Nhóm sinh viên thực hiện!

# I.Dẫn nhập và khái niệm :

## 1. Dẫn nhập:

Chúng ta thấy rằng CPU có thể được dùng chung bởi nhiều process. Do kết quả định thời CPU, chúng ta có thể cải tiến hiệu suất của CPU lẫn tốc độ đáp ứng của người dùng. Để thực hiện việc làm tăng hiệu quả này chúng ta phải lưu giữ vài quá trình trong bộ nhớ, tức là chúng ta cần phải dùng bộ nhớ dùng chung.

Bộ nhớ là trung tâm hoạt động của hệ thống máy tính hiện đại. Bộ nhớ gồm một dãy lớn của các words hoặc các byte mà mỗi cái đó đều có địa chỉ của riêng chúng.

* Quản lí bộ nhớ là công việc của HĐH với sự hỗ trợ của phần cứng nhằm phân phối, sắp xếp các process trong bộ nhớ sao cho hiệu quả.
* Mục tiêu cần đạt được là nạp càng nhiều process vào bộ nhớ càng tốt (gia tăng mức độ đa chương).
* Trong hầu hết các hệ thống, Kernel sẽ chiếm một phần cố định của bộ nhớ, phần còn lại phân phối cho các process.

## 2. Khái niệm:

- Địa chỉ luận lí hay còn gọi là địa chỉ ảo ( Virtual Address ) : là tất cả các địa chỉ do bộ vi xử tạo ra. Tập hợp tất cả các địa chỉ luận lí tạo nên không gian địa chỉ luận lí.

- Địa chỉ vật lý hay còn gọi là địa chỉ thực : là địa chỉ thực tế mà trình quản lý bộ nhớ nhìn thấy và thao tác. Tập hợp tất cả các địa chỉ vật lý tạo nên không gian địa chỉ vật lý.

- Paging & Page File : Paging là kỹ thuật được sử dụng bởi hệ thống bộ nhớ ảo để đảm bảo rằng dữ liệu của chúng ta cần là tồn tại ( available ) càng nhanh càng tốt. HĐH copy một số trang nhất định từ thiết bị lưu trữ vào bộ nhớ chính. Khi chương trình cần một trang mà hiện tại không tồn tại trong bộ nhớ chính, HĐH sẽ copy trang cần thiết đó vào bộ nhớ và copy trang khác vào lại ổ đĩa. Page File là một file trên ổ cứng, được Windows sử dụng làm bộ nhớ ảo để lưu trữ các chương trình và dữ liệu, khi bộ nhớ vật lý ( RAM ) không đủ chỗ chứa.

# II.Hệ thống Windows quản lí bộ logic theo cấu trúc phân trang (paging) :

## 1. Physical Storage :

Mức tối đa của dung lượng bộ nhớ vật lý được hệ thống Windows hỗ trợ khoảng từ 2GB- 2TB, tùy thuộc vào phiên bản của Windows .

**\*Windows 7:**

|  |  |  |
| --- | --- | --- |
| Phiên bản | Hệ 32 bit | Hệ 64 bit |
| Windows 7 Home Basic | 4 GB | 8 GB |
| Windows 7 Home Premium | 4 GB | 16 GB |
| Windows 7 Professional | 4 GB | 192 GB |
| Windows 7 Enterprise | 4 GB | 192 GB |
| Windows 7 Ultimate | 4 GB | 192 GB |

**\*Windows 8:**

|  |  |  |
| --- | --- | --- |
| Phiên bản | Hệ 32 bit | Hệ 64 bit |
| Windows 8 | 4 GB | 128 GB |
| Windows 8 Professional | 4 GB | 512 GB |
| Windows 8 Enterprise | 4 GB | 512 GB |

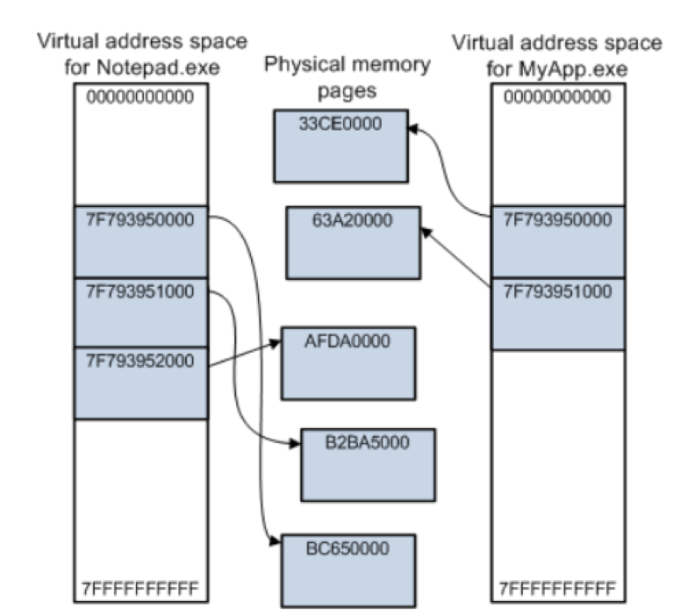
**\*Windows 10:**

|  |  |  |
| --- | --- | --- |
| Phiên bản | Hệ 32 bit | Hệ 64 bit |
| Windows 10 Home | 4 GB | 128 GB |
| Windows 10 Pro | 4 GB | 2 TB (> 2000 GB) |
| Windows 10 Education | 4 GB | 2 TB |
| Windows 10 Enterprise | 4 GB | 2 TB |

( Bảng so sánh sự giới hạn bộ nhớ vật lí ở các phiên bản khác nhau của HĐH Windows)

## 2. Virtual Address Space (Không gian địa chỉ ảo):

Virtual Address Space (Không gian địa chỉ ảo) ứng với một chương trình là một miền không gian bộ nhớ ảo được đánh địa chỉ mà tiến trình đó có thể sử dụng. Không gian địa chỉ này cho mỗi process là riêng biệt và không thể được sử dụng bởi một chương trình khác trừ khi nó được chia sẻ. Một địa chỉ ảo không diễn tả một địa chỉ vật lý thật của đối tượng ở trên vùng nhớ. Thay vào đó, hệ thống sẽ duy trì một bảng được gọi là page table cho mỗi chương trình. Bảng này cho phép ta chuyển địa chỉ ảo sang một địa chỉ vật lý thật. Mỗi khi một tiến trình con của một chương trình tham chiếu tới một địa chỉ, thì hệ thống sẽ tự động chuyển từ địa chỉ ảo sang địa chỉ thật để xử lý.



*Hình ảnh: Mô tả quá trình truyền địa chỉ ảo của một chương trình khi được tham chiếu tới sáng địa chỉ vật lý thật (quá trình thực hiện bởi hệ điều hành)*

Bảng sau đây cho thấy phạm vi bộ nhớ mặc định cho mỗi phân vùng cho windows 32 bit:

| **Dải bộ nhớ** | **Cách sử dụng** |
| --- | --- |
| 2GB thấp (0x00000000 đến 0x7FFFFFFF) | Được sử dụng bởi quá trình. |
| Cao 2GB (0x80000000 đến 0xFFFFFFFF) | Được sử dụng bởi hệ thống. |

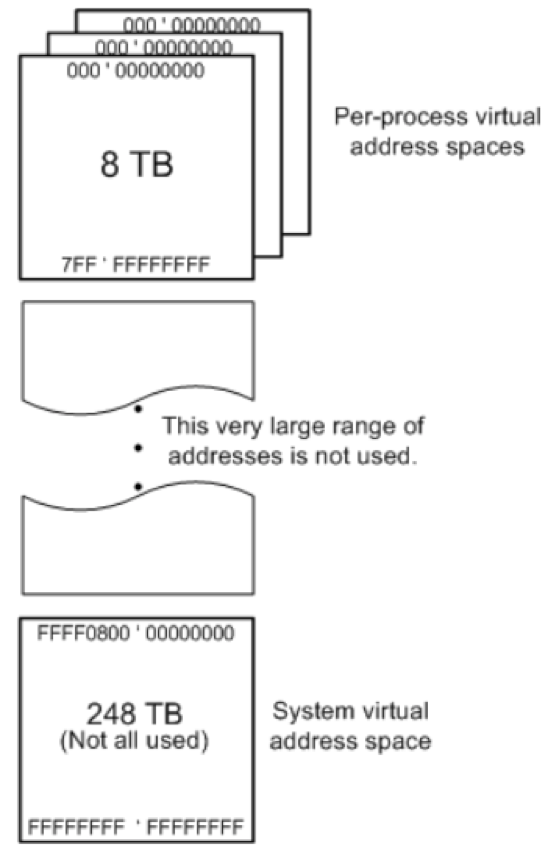
**\*Chế độ không gian địa chỉ ảo trong Windows OS – 32 bit:**

Không gian địa chỉ ảo dành cho Windows 32-bit có kích thước 4 gigabyte (GB) và được chia thành hai phân vùng: một phân vùng để quy trình sử dụng và phân vùng còn lại dành cho hệ thống sử dụng.

Đối với một chương trình chạy trên Windows OS – 32 bit, không gian địa chỉ ảo thường là 2GB trong khoảng từ địa chỉ 0x00000000 tới 0x7FFFFFFF. Trong vùng nhớ này được chia thành nhiều page, mỗi page có độ lớn thường là 1000H bytes tương ứng với 4 KB về kích thước vùng nhớ. Trong Windows OS 32 bit, tổng không gian địa chỉ ảo là 4GB. Thông thường thì 2GB địa chỉ thấp được sử dụng cho chương trình, còn lại 2GB địa chỉ cao được sử dụng cho hệ thống. Ngoài ra, trong Windows OS 32 bit có một tùy chọn ( option ) cho phép tăng kích thước không gian địa chỉ lên cho chương trình sử dụng đó là **increaseuserva –**boot entry option. Chế độ này được gọi là chế độ 4GT-( 4-gigabyte Tuning ) của hệ điều hành. Khi đó, không gian địa chỉ ảo dành cho chương trình sẽ được tăng lên tối đa là 3GB.

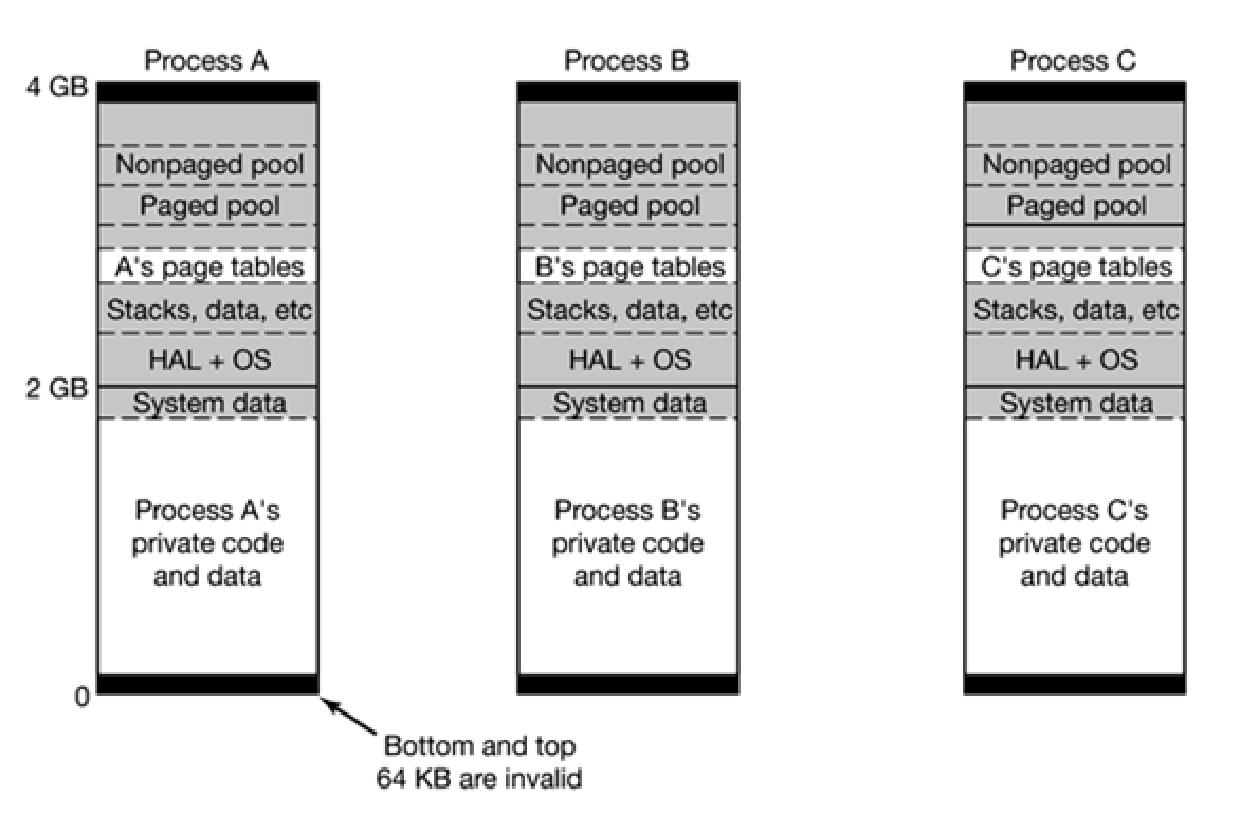
**\*Chế độ không gian địa chỉ ảo trong Windows OS – 64 bit:**

Trong Windows OS 64 bit, về lý thuyết chúng ta sẽ có một trong gian địa chỉ ảo là 2^64 bytes tương ứng với 16 exabytes, nhưng thực chất chỉ có một lượng nhỏ địa chỉ trong số 16 exabytes này được sử dụng.



* Đối với quy trình 32 bit, không gian địa chỉ ảo thường là dải 2 gigabyte từ 0x00000000 đến 0x7FFFFFFF.
* Đối với quy trình 64 bit trên Windows 64 bit, không gian địa chỉ ảo là dải 128 terabyte từ 0x000'00000000 đến 0x7FFF'FFFFFFFF.

Ngoài ra chúng ta cần chú ý tới một điều là những đoạn mã được viết ở User Mode chỉ có thể xử lý địa chỉ trong không gian của User Mode mà không được quyền xử lý tới những địa chỉ ở Kernel Mode. Hạn chế này nhằm bảo vệ user-mode code xử lý đọc hoặc ghi nhầm vào những cấu trúc dữ liệu của hệ thống. Trong khi đó, kernel-mode code có thể xử lý được những địa chỉ trong User Mode lẫn trong Kernel Mode.

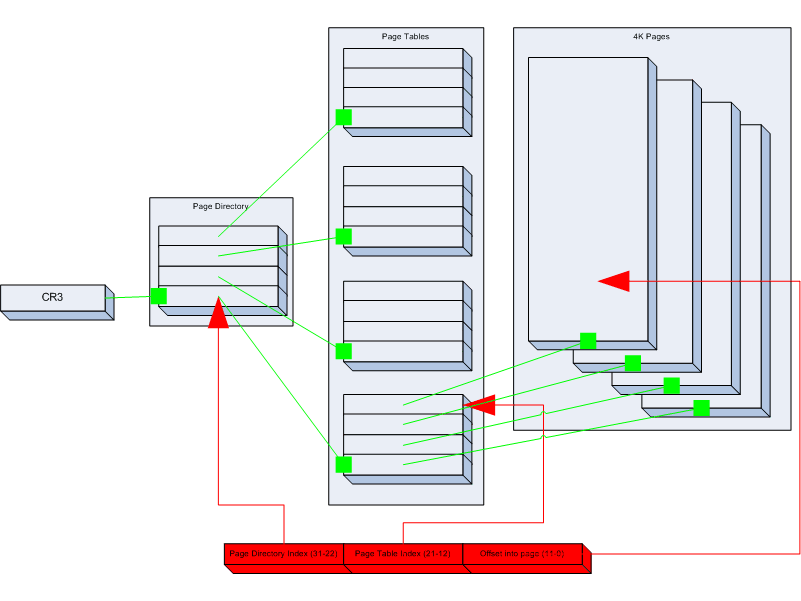
*Hình ảnh: Không gian địa chỉ ảo được cấp phát cho 3 tiến trình*

## 3. Phân trang ( paging ):

Phân trang là một hệ thống cho phép mỗi quá trình nhìn thấy một không gian địa chỉ ảo đầy đủ, mà không thực sự yêu cầu phải có hoặc hiện tại đầy đủ dung lượng bộ nhớ vật lý. Phân trang được thực hiện thông qua việc sử dụng [Đơn vị quản lý bộ nhớ](https://wiki.osdev.org/Memory_Management_Unit) (MMU). Trên x86, MMU ánh xạ bộ nhớ thông qua một loạt [bảng](https://wiki.osdev.org/Page_Tables) , chính xác là hai bảng. Chúng là thư mục phân trang (Page Directory) và bảng phân trang (Page Table).

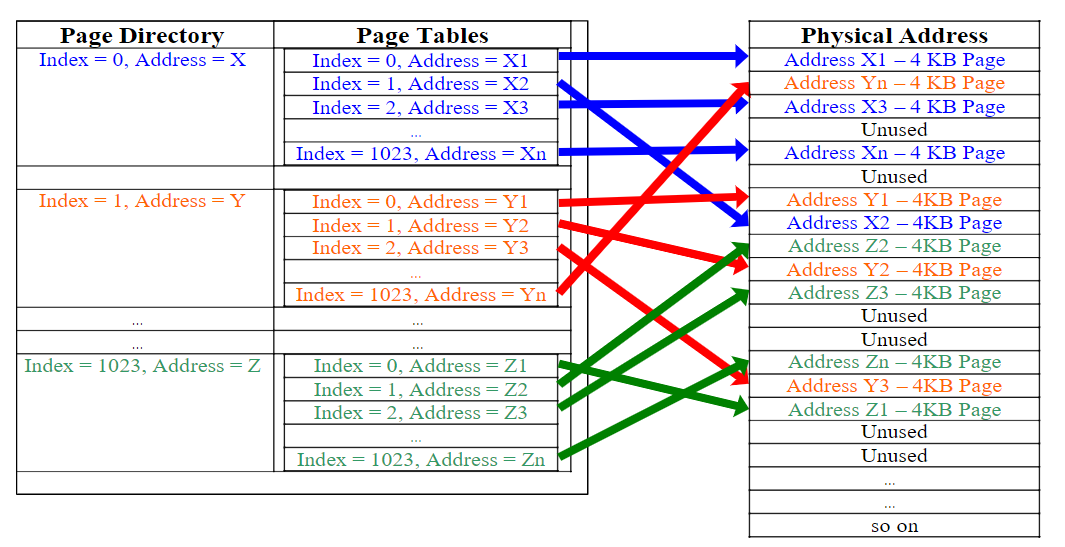
Cả hai [bảng](https://wiki.osdev.org/Page_Tables) đều chứa 1024 mục nhập 4 byte, mỗi mục có 4 KiB. Trong thư mục trang, mỗi mục nhập trỏ đến một bảng trang. Trong bảng trang, mỗi mục nhập trỏ đến một khung trang vật lý 4 KiB. Ngoài ra, mỗi mục nhập có các bit kiểm soát tính năng bảo vệ truy cập và bộ nhớ đệm của cấu trúc mà nó trỏ tới. Toàn bộ hệ thống bao gồm một thư mục trang và các bảng trang đại diện cho một bản đồ bộ nhớ ảo 4-GiB tuyến tính.

Việc dịch địa chỉ ảo thành địa chỉ vật lý trước tiên bao gồm việc chia địa chỉ ảo thành ba phần: 10 bit quan trọng nhất (bit 22-31) chỉ định chỉ mục của mục nhập thư mục trang, 10 bit tiếp theo (bit 12-21) chỉ định chỉ mục của mục nhập bảng trang và 12 bit quan trọng nhất (bit 0-11) chỉ định độ lệch trang. Sau đó, MMU đi qua các cấu trúc phân trang, bắt đầu với thư mục trang và sử dụng mục nhập thư mục trang để định vị bảng trang. Mục nhập bảng trang được sử dụng để định vị địa chỉ cơ sở của khung trang vật lý và phần bù trang được thêm vào địa chỉ cơ sở vật lý để tạo ra địa chỉ vật lý. Nếu bản dịch không thành công vì lý do nào đó (ví dụ: mục nhập được đánh dấu là không có), thì bộ xử lý sẽ gây ra lỗi trang.



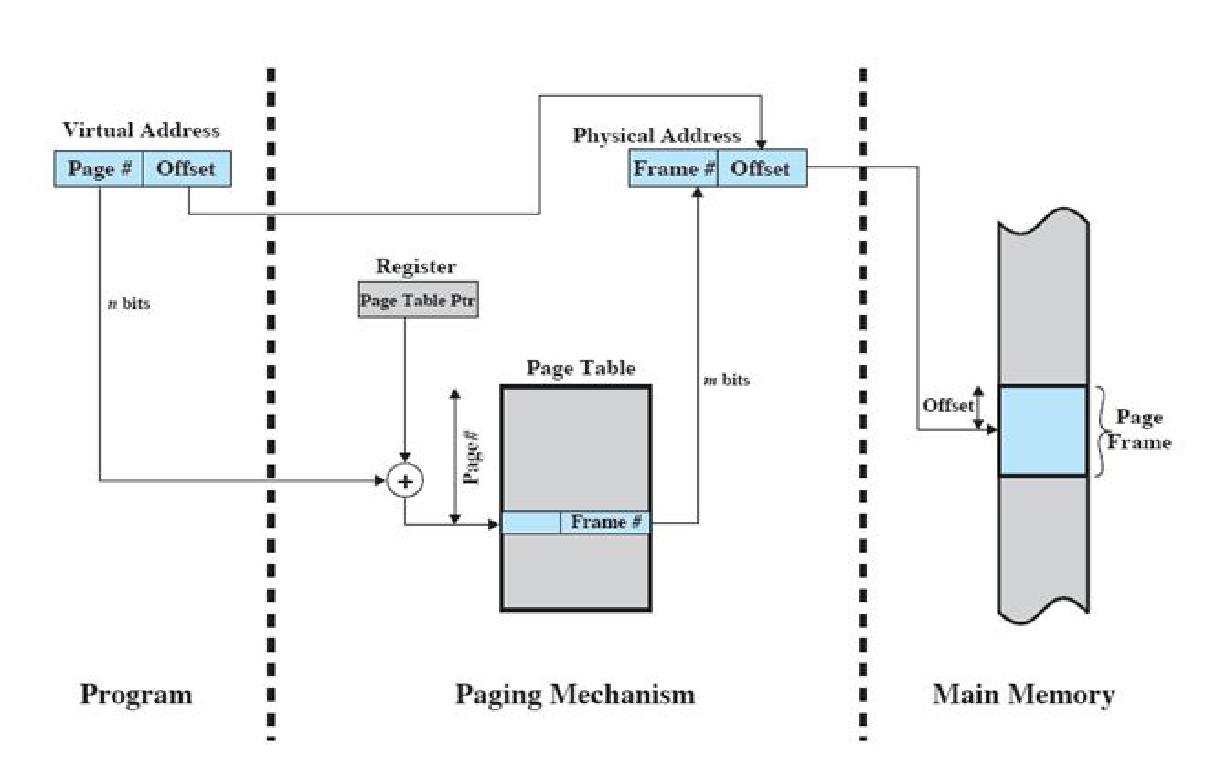
*Hình ảnh: Cấu trúc phân trang x86*

Bộ xử lý x86 chia không gian địa chỉ vật lý (hoặc bộ nhớ vật lý) trong các trang 4 KB. Vì vậy, để giải quyết 4GB bộ nhớ, chúng tôi sẽ cần 1 Mega (1024 × 1024) trang 4KB. Bộ xử lý sử dụng cấu trúc hai cấp để tham chiếu đến các trang 1 Mega này. Bạn có thể coi nó như một ma trận hai chiều gồm 1024 × 1024 phần tử. Thứ nguyên đầu tiên được gọi là Page Directory và thứ nguyên thứ hai được gọi là Page Table. Như vậy chúng ta có thể tạo 1 thư mục Trang với 1024 mục nhập, mỗi mục trỏ đến một Bảng Trang. Điều này sẽ cho phép chúng tôi có 1024 bảng trang. Mỗi bảng trang lần lượt có thể có 1024 mục, mỗi mục trỏ đến một trang 4 KB. Về mặt đồ họa, nó trông giống như sau:



*Hình ảnh: Paging in x86 Processor*

Mỗi thành phần Page Directory Entry (PDE) có kích thước 4 bytes và trỏ đến một Page Table. Tương tự , mỗi Page Table Entry (PTE) có kích thước 4 bytes và trỏ đến một physical Address (địa chỉ vật lý) của 4KB page. Để chứa 1024 PDE mà mỗi thành phần lại chứa 1024 PTE, chúng ta cần tổng bộ nhớ là 4x1024x1024 bytes, có nghĩa là 4MB. Vì vậy chia tòan bộ 4GB vùng Address (Address) cho 4KB page, chúng ta cần 4MB vùng nhớ

=> Do đó, để chia tất cả 4GB không gian bộ nhớ thành các page,ta cần 4MB bộ nhớ.Khi 1 PDE hay 1PTE được sử dụng,20bits đầu định số hiệu trang và 12bits sau để chứ thông tin trang, 20bits đầu gọi là Page Frame Number(PFN)

*Hình ảnh: Chuyển đổi địa chỉ trong phân trang (paging)*

Sự chuyển đổi địa chỉ:

+ Đầu tiên,ta đổi địa chỉ ảo chuyển thành mã nhị phân

+ Tiếp theo, do kích thước trang là 4096 (4KB) = 2^12 bytes, như vậy 20 bit đầu của địa chỉ logic sẽ là số hiệu trang, 12 bit còn lại là offset.

+ Đối chiếu số hiệu trang với bảng trang ta được số hiệu khung trang trên địa chỉ vật lý.

+ Kết hợp số hiệu khung trang nhận được từ bước trên với 12 bit offset từ mã nhị phân bên trên ta được địa chỉ vật lý.

## 4. Windows Page Table Management (Quản lý bảng trang Windows):

Trong Windows, mỗi process có Page Directory và Page Table của chính nó. Vì vậy Windows cấp 4MB của vùng space này cho mỗi process. Khi một process được cài đặt, mỗi thành phần trong Page Directory chứa Address vật lý (physical Address) của Page Table.

Các thành phần trong Page Table hoặc là valid (hợp lệ) , hoặc là invalid (ko hợp lệ). Các thành phần valid chứa physical Address của 4KB page cấp cho process. Một thành phần invalid (ko hợp lệ) chứa một vài bits đặc biệt đánh dấu nó ko hợp lệ và các thành phần này được biết như Invalid PTEs. Khi memory được cấp cho process,các thành phần trong Page Table được lắp các Address vật lý của các pages đã cấp. Bạn nên nhớ một điều ở đây là một process ko biết bất kỳ gì về Address vật lý và nó chỉ sử dụng logical Address (địa chỉ luân lý).Chi tiết về việc logical Address nào tương ứng với physical Address nào được quản lý chuyển đổi bởi Windows Memory Manager và Processor (bộ vi xử lý).

Address tại Page Directory nào của một process được định vị trong physical memory được tham chiếu đến như là Page Directory Base Address. Page Directory Base Address này được chứa trong một thanh ghi đặc biệt của CPU là CR3 (trên nền x86). Để chuyển đổi context khác, Windows tải một giá trị mới của CR3 để trỏ đến một Page Directory base mới của process. Với cách này mỗi process sẽ lấy được các phần phân chia cả 4GB physical Address space của chính nó. Tất nhiên tổng memory cấp tại một thời điểm cho tất cả các process trong hệ thống là ko thể vượt quá số lượng RAM+ kích thước pagefile nhưng theo lược đồ đã thảo luận ở trên thì cho phép Windows cấp cho mỗi process vùng Address logical (hay Virtual: ảo) 4GB . Chúng ta gọi nó là vùng địa chỉ ảo (Virtual Addresses sapce) bởi vì ngay mỗi process có đến cả range (phạm vi) là 4GB Address , nó chỉ có thể sử dụng memory cấp cho nó.

Nếu một process thử truy xuất (access) một Address ko được cấp phép, nó sẽ gây ra một access violation (sự vi phạm truy xuất) bởi vì PTE tương ứng với Address trỏ đến một giá trị ko hợp lệ (invalid value). Cũng vậy, process ko thể cấp memory nhiều hơn những gì nó được phép trong system. Phương thức tách riêng logical memory từ physical memory này có nhiều thuận lợi. Một process có được một vùng Address 4GB tuyến tính , do đó các lập trình viên ứng dụng ko còn phải lo lắng về segments và hòan tòan ko giống như những ngày tháng cũ làm việc với DOS. Nó cũng cho phép Windows run nhiều prosses cùng một lúc và cho phép chúng dùng physical memory trên máy tính mà ko phải lo lắng về chúng sẽ đè lên trên vùng Address space của process khác. Một logical Address trong một process sẽ ko bao giờ trỏ đến một physical memory được cấp cho process khác (trừ khi chúng sử dụng phần nào để shared memory). Vì vậy,một process có thể ko bao giờ read hay write vào memory của process khác.

Sự chuyển đổi từ logical Address sang physical Address được thực hiện bởi bộ vi xử lý. Một 32bit logical Address được chia thành 3 phần như hình dưới đây.

|  |  |  |
| --- | --- | --- |
| 10 bits | 10 bits | 12 bits |

Vi xử lý loads physical Address của page directory base lưu trử trong CR3. Rồi nó được sử dụng 10 bits thấp từ logical Address như là một chỉ mục trong Page directory. Điều này cho processor một page directory entry (PDE) trỏ đến một Page Table. 10 bits kế đến được sử dụng như một chỉ mục trong Page Table. Sử dụng 10 bits này, nó lấy một page table entry (hay PTE) trỏ đến một 4KB physical page. 12 bits thấp nhất được sử dụng đánh Address các bytes riêng lẻ trên một page.

## 5. Windows Memory Protection (Bảo vệ bộ nhớ Windows):

Chức năng bảo vệ bộ nhớ này có thể không cho phép một tiến trình đọc hoặc ghi lên vùng nhớ không được cấp phát cho nó, ngăn các phần mềm độc hại hoặc các đoạn mã lỗi trong chương trình can thiệp vào quá trình xử lý của một chương trình khác.

Bộ nhớ thuộc về một tiến trình được bảo vệ ngầm bởi không gian địa chỉ ảo riêng của nó. Ngoài ra, Windows cung cấp khả năng bảo vệ bộ nhớ bằng cách sử dụng phần cứng bộ nhớ ảo. Việc triển khai bảo vệ này thay đổi theo bộ xử lý, ví dụ: các trang mã trong không gian địa chỉ của một quy trình có thể được đánh dấu là chỉ đọc và được bảo vệ khỏi sửa đổi bởi các luồng chế độ người dùng.

Windows hổ trợ sự bảo vệ memory cho tất cả các processes mục đích để một process ko thể truy xuất một vùng memory của process khác. Điều này đảm bảo các họat động của nhiều processes cùng 1 lúc 1 cách trôi trãi. Windows đảm bảo chế độ bảo vệ này bằng cách theo các bước sau:

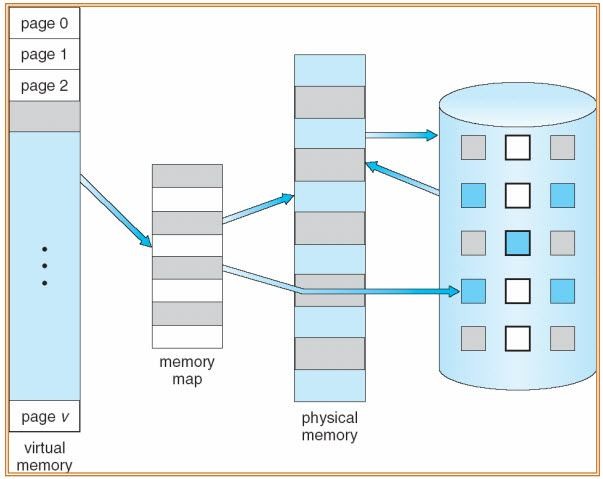
+ Nó chỉ put physical address của memory được định vị trong PTE cho một process. Điều này đảm bảo rằng process bắt được một access violation nếu nó thử truy xuất một addr mà ko được định vị.

+ Một rouge process (tiến trình đang thực thi) có thể thử thay đổi page tables của nó mục đích là nó có thể truy xuất physical memory thuộc về một process khác.Windows protect lọai tấn công này bởi cất giữ page tables trong kernel address space. Trở lại thảo luận vấn đề của chúng ta lúc đầu, đó là vấn đề : ngòai phạm vi 4GB logical addr space được cấp cho 1 process, 2GB được cấp cho user mode và 2GB được trữ cho windows kernel. Vì vậy một user mode app (ứng dụng mode người dùng) ko có thể truy xuất trực tiếp hoặc thay đổi page tables. Tất nhiên nếu một kernel mode driver muốn làm điều đó, nó có thể làm được bởi một khi bạn ở trong kernel mode, bạn hầu như làm chủ tòan hệ thống

# III. Quản lí bộ nhớ ảo ( bộ nhớ Logic ) :

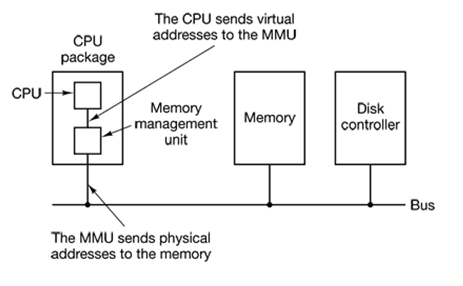
## 1. Tổ chức bộ nhớ ảo

Bộ nhớ ảo ( Virual Memory ): là một kĩ năng cho phép xử lí 1 tiến trình không được nạp toàn bộ vào bộ nhớ vật lí. Bộ nhớ ảo là mô hình hóa bộ nhớ như 1 bảng lưu trữ rất lớn và đồng nhất, tách biệt hẳn khái niệm không gian địa chỉ ảo (virtual address space) và không gian vật lý (physical space). Một điểm lợi quan trọng của cơ chế này là các chương trình được chạy có thể lớn hợn bộ nhớ vật lý. Ngoài ra, bộ nhớ ảo phóng đại bộ nhớ chính thành bộ nhớ luận lý cực lớn khi được hiển thị bởi người dùng. Kỹ thuật này giải phóng người lập trình từ việc quan tâm đến giới hạn kích thước bộ nhớ. Bộ nhớ ảo cũng cho phép các quá trình dễ dàng chia sẻ tập tin và không gian địa chỉ, cung cấp cơ chế hữu hiện cho quá trình.



*Hình ảnh: Minh họa bộ nhớ ảo lớn hơn bộ nhớ vật lý*

Để thi hành một lệnh nào đó, CPU gửi địa chỉ ảo đến Bộ phận dịch (Memory Management Unit - MMU). Thông qua MMU, địa chỉ ảo này sẽ được ánh xạ tương ứng với một địa chỉ vật lý cụ thể và được gửi tới bus địa chỉ. Cuối cùng thông qua bus địa chỉ để truy cập tới 1 vùng nhớ cụ thể trên RAM.



*Hình ảnh: CPU làm việc với MMU*

Trình quản lý bộ nhớ ảo (Virtual Memory Manager - VMM) của hệ điều hành quản lý các yêu cầu bộ nhớ được thực hiện bởi tất cả các quá trình. VMM phân bổ địa chỉ liên tiếp trong bộ nhớ ảo theo đơn vị được gọi là Trang (Page).

Ánh xạ của Trang ảo đến bộ nhớ vật lý xảy ra trong các đơn vị được gọi là khung trang (**Page frame)**

Trang là một thực thể ảo và do đó không gian bộ nhớ của nó không tồn tại vật lý. Trang chỉ tồn tại ở dạng địa chỉ trang trên Bảng trang là một thực thể vật lý. Bảng Trang này về cơ bản là Bộ nhớ ảo và một Trang tương tự như một vé trong bộ nhớ ảo. Bản sao của Trang là Mảng Khung Trang trong Bộ nhớ vật lý và là danh tính thực của Trang.

Do đó hệ điều hành duy trì một Bảng trang (**Page Table)** cho mỗi tiến trình, chứa các trang tiến trình trong bộ nhớ ảo được ánh xạ tới khung trang của nó trong bộ nhớ vật lý hoặc được khởi tạo vào vị trí trang trên đĩa cứng.

Bảng Trang là một tập hợp các địa chỉ Trang trong một mảng, trong đó các chỉ mục của mảng đại diện cho địa chỉ trang ảo và các giá trị mảng được gọi là Mục nhập Bảng Trang (Page Table Entry - PTE) chứa địa chỉ khung trang tương ứng trong bộ nhớ vật lý. Bảng trang của tất cả các tiến trình đang chạy được tạo trên bộ nhớ vật lý khi khởi động hệ thống.

## 2. Không gian địa chỉ ảo (virtual address space)

Không gian địa chỉ ảo cho một tiến trình là tập hợp các địa chỉ bộ nhớ ảo mà nó có thể sử dụng. Không gian địa chỉ cho mỗi quá trình là riêng tư và không thể được truy cập bởi các quá trình khác trừ khi nó được chia sẻ.

Địa chỉ ảo không đại diện cho vị trí vật lý thực của một đối tượng trong bộ nhớ; thay vào đó, hệ thống duy trì một bảng trang cho mỗi quá trình, đây là một cấu trúc dữ liệu nội bộ được sử dụng để dịch các địa chỉ ảo thành địa chỉ vật lý tương ứng của chúng. Không gian địa chỉ ảo cho Windows 32-bit có kích thước 4 gigabyte (GB) và được chia thành hai phân vùng: một phân vùng để sử dụng cho tiến trình và phân vùng còn lại để hệ thống sử dụng. Không gian địa chỉ ảo cho Windows 64-bit có kích thước vài terabyte. Không gian địa chỉ ảo dduojc chia thành 2 thành phần Sô trang ảo (Virtual page number – VPN) và OFFSET.

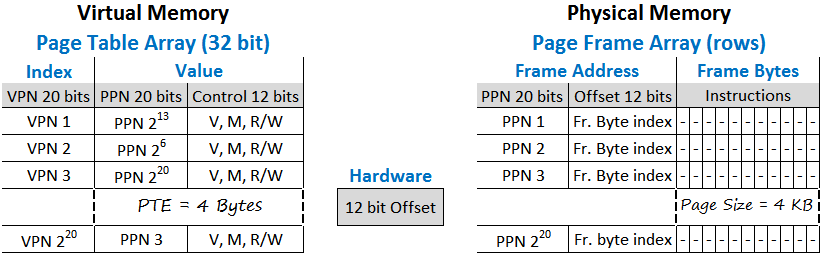
Ví dụ với không gian đia chỉ ảo 32-bit:

Không gian địa chỉ 32 bit = **Số trang ảo (VPN) 20 bit + OFFSET** 12 bit

Trong đó VPN 20 bit được sử dụng làm địa chỉ của mỗi mảng và OFFSET 12 bit được sử dụng làm chỉ mục (vị trí) để xác định các phần tử trong mảng. Như vậy, 2VPN = Số mảng (Pages), 2OFFSET = Số phần tử của mảng (bytes).

Khi mảng Khung trang được tạo trong Bộ nhớ vật lý, mảng này được gán với địa chỉ mảng 20 bit được gọi là Số trang vật lý **(Physical Page Number – PPN). Trong khi VPN là các chỉ mục liên tiếp của một bảng trang, thì PPN cũng là địa chỉ khung liên tiếp trong bộ nhớ vật lý.**

**\*Địa chỉ của vị trí bộ nhớ vật lý được hình thành bằng cách ghép PPN với Offset.**



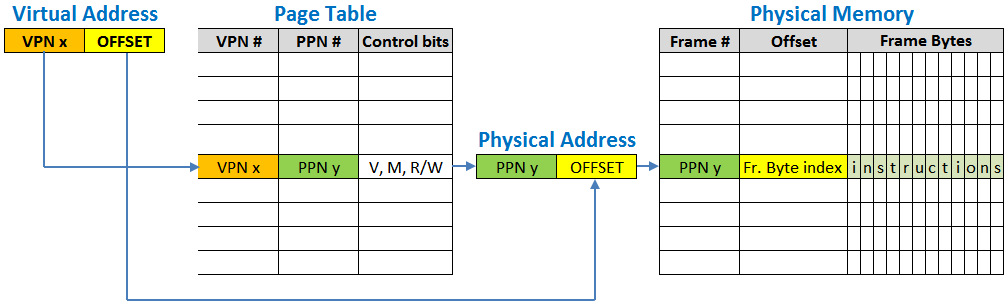
## 3. Phân trang nhu cầu

Phân trang theo yêu cầu là một kỹ thuật quản lý bộ nhớ dựa trên các nguyên tắc độc lập với kiến ​​trúc hệ thống hoặc RAM được cài đặt và áp dụng cho Mở rộng địa chỉ vật lý (**Physical Address Extension** – PAE) cũng như các hệ thống 64-bit.

Hệ điều hành chịu trách nhiệm quản lý tất cả thông tin liên quan đến một quy trình trong Khối điều khiển quy trình (Process Control Block - PCB) trong nhân hệ điều hành. Chúng được phân loại rộng rãi thành dữ liệu nhận dạng quy trình, dữ liệu trạng thái quy trình và dữ liệu kiểm soát quy trình. Dữ liệu nhận dạng quy trình ánh xạ giữa ID quy trình và Địa chỉ cơ sở của bảng trang (Page Table Base Address - PTBA) là địa chỉ bắt đầu của một Bảng trang trong bộ nhớ vật lý. Mỗi CPU đều có Thanh ghi cơ sở bảng trang (Page Table Base Register - PTBR) mà chỉ hệ điều hành mới có thể truy cập được để lưu trữ địa chỉ cơ sở bảng trang hiện tại. Trong quá trình chuyển đổi, hệ điều hành chỉ cần thực hiện thay đổi địa chỉ cơ sở của bảng trang trong PTBR.

CPU có Bộ đếm chương trình chứa địa chỉ ảo thành 2 phần. Các bit bậc cao chứa VPN là bộ đếm tăng theo số trang ảo và các bit bậc thấp chứa Offset là bộ đếm byte để lập chỉ mục vào các phần tử bộ nhớ trong một trang. Các VPN là giống nhau đối với tất cả các Bảng trang vì địa chỉ bộ nhớ ảo được tạo bởi Bộ đếm chương trình là giống nhau đối với tất cả các quá trình. Bộ đếm chương trình này được điều khiển bởi bộ logic trong phần cứng CPU.

Đầu tiên Bộ quản lý Bộ nhớ (MMU) của CPU đề cập đến địa chỉ Cơ sở Bảng Trang trong PTBR để truy cập Bảng Trang trong bộ nhớ vật lý, sau đó tham chiếu đến phần VPN của địa chỉ ảo trong Bộ đếm Chương trình để lập chỉ mục vào Bảng Trang và xác định PPN tương ứng với VPN. MMU tạo địa chỉ vật lý bằng cách nối PPN với phần Offset của địa chỉ ảo trong Bộ đếm chương trình và lưu trữ địa chỉ vật lý này trên Thanh ghi địa chỉ bộ nhớ (Memory Address Register - MAR) của CPU. CPU đọc địa chỉ trong MAR để lấy lệnh từ bộ nhớ vật lý và thực thi nó. Chênh lệch trong Bộ đếm chương trình tăng lên để trỏ đến vị trí bộ nhớ ảo tiếp theo và chu kỳ lặp lại. Khi Offset vượt qua giá trị lớn nhất của nó (tất cả 1s), phần VPN trong Bộ đếm chương trình sẽ được tăng lên để trỏ đến VPN tiếp theo trong Bảng trang.



Khi tất cả các khung Trang được tạo và quá trình sẵn sàng để thực thi, Hệ điều hành sẽ cập nhật PTBR của CPU với địa chỉ Cơ sở Bảng Trang và đặt lại Bộ đếm Chương trình. MMU lập chỉ mục vào Bảng Trang và chỉ tải MAR nếu PPN có bit Hợp lệ = 1.

Nếu một lệnh được tìm nạp là một con trỏ đến một vị trí bộ nhớ khác có chứa lệnh thực, thì MAR được cập nhật vào vị trí bộ nhớ của địa chỉ con trỏ. Hình ảnh của PTBR, Bộ đếm chương trình và tất cả các thanh ghi CPU được lưu trong bộ nhớ trước khi việc thực thi chuyển đến vị trí bộ nhớ được cập nhật trong MAR. Sau khi hoàn thành (các) lệnh, câu lệnh trả về tải lại các thanh ghi có hình ảnh đã lưu để việc thực thi trở về địa chỉ ảo ban đầu và tiếp tục hoạt động tuần tự. Điều này được gọi là chuyển đổi ngữ cảnh (**Context switching)**.

## 4. Page Fault

MMU tạo ra lỗi trang (Page Faults) nếu địa chỉ trang ảo bị thiếu số khung trong bảng trang Hoặc số khung không tồn tại trong bộ nhớ vật lý Hoặc địa chỉ vật lý không phải là một phần của bộ làm việc hiện tại mà nằm trong danh sách chờ / sửa đổi.

Windows gọi trình xử lý trang (**Page handler**) của cố gắng tìm cách giải quyết tùy thuộc vào loại lỗi của trang. Trình xử lý Trang trước tiên đảm bảo rằng yêu cầu trang là chính hãng. Nếu nó được phát hiện là không chính hãng, quá trình sẽ bị chấm dứt. Nếu đó là một yêu cầu chính hãng, hệ điều hành sẽ xác định mã / dữ liệu bị thiếu và tải nó từ đĩa cứng vào một vị trí trống trong bộ nhớ, do đó tạo ra một khung trang mới. Địa chỉ khung Trang mới được cập nhật trên trang bị lỗi trong Bảng Trang và trang được đánh dấu là hợp lệ với bit hợp lệ = 1. Nếu khung Trang ở trong bộ nhớ chờ / đã sửa đổi, MMU chỉ cần cập nhật số khung trên trang lỗi sẽ đưa trang trở lại trạng thái chờ về chế độ làm việc. Trình xử lý trang cuối cùng quay trở lại quy trình ban đầu tiếp tục thực hiện lại trang lỗi.

## 5. Bộ đệm giao diện dịch (Translation Lookaside Buffer - TLB)

Bộ đệm giao diện dịch (TLB) là bộ đệm được duy trì trên các thanh ghi của MMU của bộ xử lý, để lưu trữ các trang được sử dụng gần đây của bảng trang. Khi MMU cần đọc một Trang từ Bảng Trang trong bộ nhớ, trước tiên nó sẽ cố gắng đọc nó từ TLB theo mặc định, nhanh hơn nhiều so với đọc từ bộ nhớ. Nếu Trang không tồn tại trong TLB, thì trước tiên MMU sao chép Trang từ bảng trang vào TLB và sau đó đọc nó từ TLB

TLB dựa trên nguyên tắc thống kê về vị trí, trong đó một trang được truy cập gần đây sẽ được yêu cầu truy cập lại trong thời gian ngắn được gọi là vị trí theo thời gian (Temporal Locality); và các số trang gần một trang được truy cập có xác suất cao để được truy cập trong một vài tham chiếu bộ nhớ tiếp theo được gọi là vị trí theo không gian (**Spatial Locality)**.

Vị trí theo thời gian được hỗ trợ bằng cách lưu vào bộ nhớ đệm của các trang được truy cập gần đây trong TLB. Trong khi vị trí theo không gian được lưu vào bộ nhớ cache tất cả các Trang hợp lệ trong Bảng trang trước khi chúng thực sự được yêu cầu.

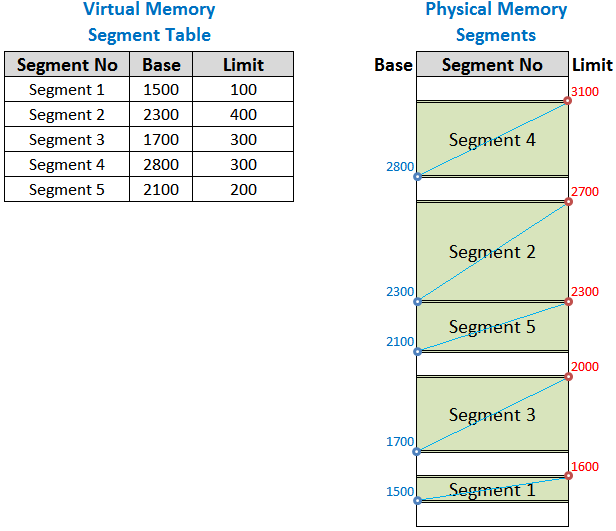
Khi một trang được tìm thấy trong TLB, nó được gọi là TLB hit và khi không tìm thấy trang, nó được gọi là TLB miss. TLB hit loại bỏ nhu cầu truy cập Bảng trang trong bộ nhớ, trong khi TLB miss phát sinh quyền truy cập Bảng Trang trong bộ nhớ.

## ****6. Phân đoạn****

Phân đoạn là một phương pháp đánh địa chỉ bộ nhớ toàn diện và có thể được thực hiện độc lập để quản lý trực tiếp bộ nhớ Vật lý.

Với sự ra đời của các hệ thống 32 bit, các chương trình được tổ chức trong bộ nhớ ảo để vượt qua giới hạn không gian do bộ nhớ vật lý áp đặt . Do đó, nhiều chương trình có thể tồn tại trên bộ nhớ ảo nhưng chỉ một vài chương trình tại một thời điểm có thể chiếm bộ nhớ vật lý.

Một chương trình là một tập hợp các phân đoạn có kích thước khác nhau.



Không gian địa chỉ ảo của một quá trình được xác định trong **Bảng Phân đoạn (Segment Table)** được giữ trong bộ nhớ liên kết trong CPU bởi **Thanh ghi Cơ sở Bảng Phân đoạn (Segment Table Base Register - STBR).**Bảng Phân đoạn chứa một tập hợp **Số phân đoạn (Segment Numbers)** cố định dưới dạng địa chỉ ảo với địa chỉ vật lý tương ứng được chỉ định bởi **Cơ sở phân đoạn (Segment Base).**

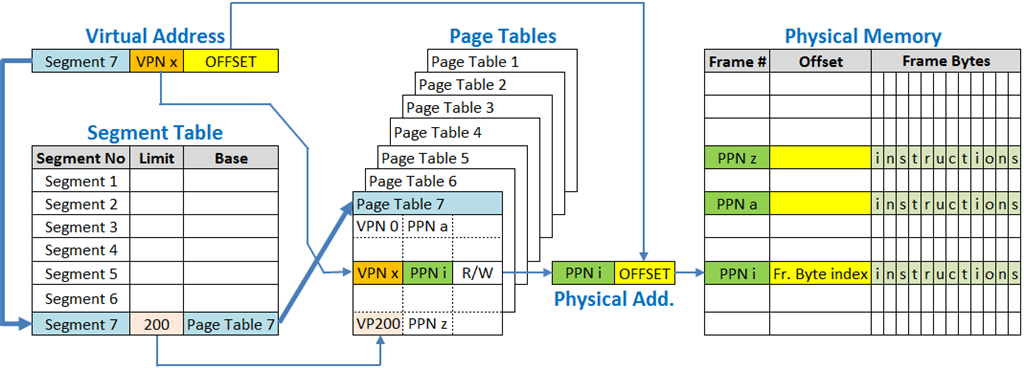
Cơ sở phân đoạn là địa chỉ bắt đầu của phân đoạn trong bộ nhớ vật lý và là tham chiếu của CPU tới một phân đoạn. Giới hạn xác định kích thước phân đoạn có thể thay đổi và phụ thuộc vào kích thước của phân đoạn được xác định bởi chương trình. Do đó, Bảng Phân đoạn xác định bộ nhớ ảo nơi mỗi số phân đoạn ảo ánh xạ tới một khối liên tục trong bộ nhớ vật lý được chỉ định bởi Địa chỉ cơ sở và Giới hạn.

Bộ đếm chương trình đặt lại và tạo địa chỉ ảo đầu tiên với các bit thứ tự cao chứa số Phân đoạn và bit thứ tự thấp chứa Chênh lệch để lập chỉ mục vào các byte bộ nhớ trong giới hạn phân đoạn hiện tại. Bộ đếm Offset kết hợp một logic để kiểm tra xem nó không vượt quá Giới hạn được xác định cho một phân đoạn nhất định. MMU đọc số phân đoạn trong Bộ đếm chương trình và tham chiếu đến bảng Phân đoạn để lấy Cơ sở phân đoạn. Địa chỉ cơ sở được nối với Offset trong Bộ đếm chương trình để tạo ra địa chỉ vật lý được lưu trữ trong MAR. CPU tìm nạp lệnh từ địa chỉ bộ nhớ trong MAR và thực hiện lệnh.

Vì số phân đoạn có thể xác định một đơn vị bộ nhớ ảo lớn hơn một trang, nên chỉ một vài số phân đoạn là đủ để tạo thành một quy trình. Kích thước Bảng phân đoạn bao gồm 5-10 số phân đoạn ảo ít hơn đáng kể so với kích thước Bảng trang bao gồm 220 số trang ảo. Tuy nhiên, ánh xạ số Phân đoạn tới một phân đoạn lớn ở các vị trí bộ nhớ vật lý liên tục sẽ gây ra Phân mảnh bên ngoài (**External Fragmentation**) dẫn đến Rò rỉ bộ nhớ (Memory Leakage), khi quá nhiều khe bộ nhớ trống bị mất giữa các phân đoạn mà độ dài liên tục không đủ để phân bổ cho một phân đoạn.

## 7. Kết hợp phân đoạn với phân trang

Thiếu sót trong phân đoạn thuần túy này đã được tìm cách cải thiện bằng cách kết hợp Phân đoạn với Phân trang theo nhu cầu trong đó mỗi phân đoạn bao gồm các trang liên tiếp được xác định trong Bảng trang với các Khung trang tương ứng được xáo trộn trong bộ nhớ vật lý . Điều này giúp loại bỏ nguồn phân mảnh bên ngoài do cấu trúc một phân đoạn vật lý ở các vị trí bộ nhớ liên tục



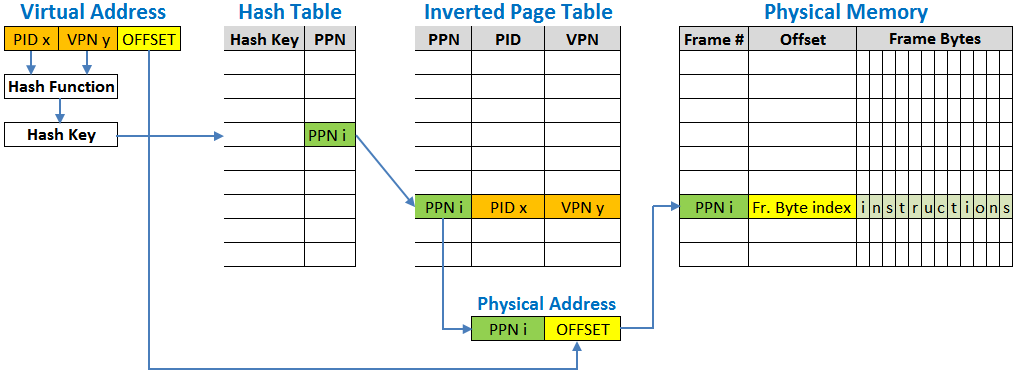
Bảng phân đoạn nằm trong thanh ghi cơ sở bảng phân đoạn (Segment Table Base Register - STBR) của CPU trong bộ nhớ liên kết trong khi Bảng trang nằm trong bộ nhớ vật lý. Cả hai thông tin bảng đều có nguồn gốc từ Khối điều khiển quá trình trong nhân hệ điều hành. Bảng Trang trên mỗi phân đoạn có số lượng trang thay đổi tùy thuộc vào kích thước của phân đoạn và do đó mỗi bảng trang có không gian địa chỉ ảo khác nhau được tạo bởi các đơn vị trang có kích thước cố định.

Một quá trình bắt đầu khi hệ điều hành tải STBR với Bảng Phân đoạn chứa các địa chỉ Cơ sở Bảng Trang. Bộ đếm chương trình đặt lại và tạo địa chỉ ảo đầu tiên. Các bit thứ tự cao nhất bao gồm bộ đếm số Phân đoạn, tiếp theo là các bit thứ tự giữa chứa VPN là bộ đếm trang ảo và các bit thứ tự thấp chứa Offset là bộ đếm byte để lập chỉ mục vào các phần tử bộ nhớ trong một trang. Bộ đếm trang ảo kết hợp logic để kiểm tra xem nó không vượt quá Giới hạn (số trang) được xác định cho Bảng trang của phân đoạn hiện đang thực thi hay không.

MMU đọc số phân đoạn trong Bộ đếm chương trình và tham chiếu đến bảng Phân đoạn để lấy địa chỉ Cơ sở Bảng Trang trong bộ nhớ vật lý. Phần VPN trong Bộ đếm chương trình sau đó được sử dụng để lập chỉ mục vào Bảng trang đó và định vị số khung vật lý (PPN). PPN được nối với Offset trong Bộ đếm chương trình để tạo ra địa chỉ vật lý được lưu trữ trong MAR để CPU tìm nạp và thực thi lệnh từ bộ nhớ.

## ****8. Bảng trang đảo ngược (Inverted Page Table)****

Bảng trang được đảo ngược là một chiến lược để giảm kích thước bảng trang bằng cách xác định bảng Trang trong đó không gian địa chỉ đề cập đến không gian địa chỉ vật lý của RAM thay vì không gian địa chỉ ảo. Điều này có ý nghĩa đối với các hệ thống máy chủ có dung lượng RAM được cài đặt lớn để đáp ứng nhu cầu của tất cả các dịch vụ đang chạy. Trong khi một Bảng Trang bình thường được VPN lập chỉ mục (định địa chỉ) tới các PTE chứa PPN, Bảng Trang Đảo ngược được PPN lập chỉ mục cho các PTE chứa VPN - do đó giới hạn không gian bộ nhớ ảo đối với không gian bộ nhớ vật lý.



Bảng Trang Đảo ngược được tạo với PPN làm chỉ mục, sau đó được cập nhật bởi PID và VPN, và cuối cùng các Khung Trang được tạo trong bộ nhớ vật lý trong chuỗi chỉ mục (PPN) của Bảng Trang Đảo ngược.

Bảng trang được đảo ngược là một mảng có chỉ mục tương ứng với Địa chỉ khung trang trong bộ nhớ vật lý (PPN) và PTE là một ánh xạ ngược tới Địa chỉ trang ảo (VPN) của quá trình sở hữu khung. Điều này không thay đổi nguyên tắc đằng sau hoạt động của bộ nhớ ảo, trong đó địa chỉ bộ nhớ ảo đóng vai trò như một con trỏ bộ nhớ trong các lệnh hệ điều hành để truy cập bộ nhớ vật lý.

# IV. Quản lí bộ nhớ vật lí:

## 1. Phân chia vùng trong RAM:

Working Set: Vùng chứa các trang đang hoạt động.

Modified Page List: Vùng chứa đã sửa đổi bị loại khỏi Working Set vì ở trạng thái rỗi quá lâu, các trang vẫn còn liên quan đến tiến trình đã gọi nó.

Standby Page List: Vùng chứa các trang chưa sửa đổi bị loại ra khỏi Working Set, còn liên quan đến tiến trình gọi nó, nhưng có 1 bản sao trên vùng Paging File ở bộ nhớ ngoài, vì vậy có thể xóa bản gốc trên RAM nếu cần.

Free Page List: Vùng chứa các trang chưa được cấp phát cho bất kỳ tiến trình nào hoặc đã được cấp phát trước đó nhưng được trả lại để sử dụng lại khi quá trình kết thúc.

Zeroed Page List: Vùng chứa các trang chuyển từ Free Page List và được khởi tạo lại bằng mã 0.

## 2. Cách thức chuyển đổi giữa các vùng trong RAM:

Giải thuật thay trang đảm bảo cho các trang ảo được nạp vào RAM khi cần và được xóa khỏi RAM khi không cần dùng nữa để thay bằng trang khác. Sơ đồ sau mô tả giải thuật thay trang.

Diagram, schematic

Description automatically generated

*Hình ảnh: Các vùng trên RAM*

Khi tiến trình gọi đến 1 trang thì nó được nạp vào vùng Working Sets.

Cứ khoảng 4s, nếu 1 trang đã sửa đổi của 1 tiến trình nào đó rỗi, nó sẽ bị đẩy từ Working Set sang đáy Modified Page List và nếu trang đó chưa sửa đổi thì sẽ bị đẩy sang đáy Standby Page List, biểu diễn bởi. Các trang ở 2 vùng này vẫn được liên kết với vùng Working Set tương ứng, có giá trị và có thế được tiến trình gọi lại và nạp vào Working Set, biểu diễn bởi. Khi tiến trình kết thúc và trang không còn chia sẻ với tiến trình nào khác, trang từ Working Set bị đẩy sang Free Page List, biểu diễn bởi.

Sau một thời gian nhất định các trang ở Modified Page List bị đẩy sang vùng Standby Page List, biểu diễn bởi. Sự khác biệt giữa 2 vùng này là Modified Page List có thể được đẩy vào Working Set nhanh hơn, còn Standby Page List có 1 bản backup ở bộ nhớ ngoài, vì vậy các trang ở Standby Page List có thể được xóa đi nếu RAM đầy, khi cần thì nạp trang backup.

Các trang ở Standby Page List khi không còn gắn với tiến trình nào nữa thì bị đẩy ra vùng Free Page List, biểu diễn bởi. Các trang ở Free Page List vẫn chứa dữ liệu nhưng đã không còn giá trị, và có thể bị ghi đè bởi 1 trang mới chuyển vào Working Set hoạt động, biểu diễn bởi.

Trong một số trường hợp đặc biệt, các tiến trình đòi hỏi các trang hoàn toàn chưa chứa dữ liệu đê ghi thông tin mới, các trang này được lấy từ vùng Zeroed Page List. Zeroed Page List có được nhờ xóa dữ liệu các trang ở Free Page List, biểu diễn bởi.

## 3. CSDL về khung trang:

Để quản lý RAM, các vùng trang trên RAM và tiện lợi cho việc ánh xạ bộ nhớ ảo vào RAM, Windows sử dụng 1 bảng dữ liệu về các khung trang. Các thông tin có thể đọc được từ bảng này gồm có:

* Số thứ tự trang trên bộ nhớ vật lý.
* Vùng mà trang đang tồn tại.
* Số bảng trang đang trỏ đến trang 1 trang.
* Con trỏ trỏ đến bảng trang đang sử dụng trang.
* Con trỏ đến trang tiếp theo trong cùng vùng List trên RAM
* …

Table

Description automatically generated

*Hình ảnh: Bảng dữ liệu khung trang*

*Nhóm 3 chúng em xin gửi lời cảm ơn chân thành nhất tới thầy !!!*