**BÁO CÁO HỆ ĐIỀU HÀNH – ASS1**

**2.2 Memory Management**

Size của virtual RAM là 1MB nên chúng ta phải sử dụng 20 bit để đại diện cho địa chỉ của mỗi byte.

Thực hiện phân đoạn với cơ chế phân trang (Paging mechanism):

* 5 bit đầu tiên đại diện cho segment index
* 5 bit kế tiếp đại diện cho page index
* 10 bit cuối cùng đại diện cho offset

Một số hàm như:

* get page table () (trong mem.c): Tìm bảng trang (page table) được cung cấp index của process.
* translate () (trong mem.c): sử dụng hàm get page table () để dịch một địa chỉ ảo sang địa chỉ vật lý.

***Note:***

Một cấu trúc đặc biệt là \_mem\_stat dùng để theo dõi trạng thái của các physical pages. Nhiệm vụ của \_mem\_stat là duy trì trạng thái (used / free) của các trang đó.

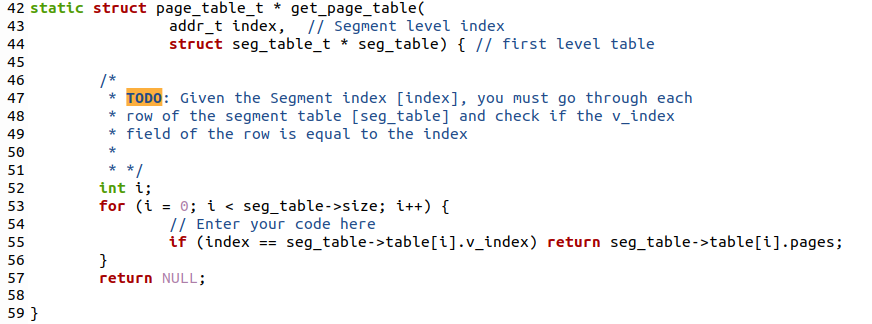
Cụ thể, \_mem\_stat là một bảng trong đó mỗi hàng là một cấu trúc được định nghĩa như sau:

****

Trong đó proc là PID của process hiện đang sử dụng trang này. Nếu proc = 0, trang này free và hệ điều hành có thể phân bổ nó cho bất kỳ quá trình nào.

**Code mem.c**

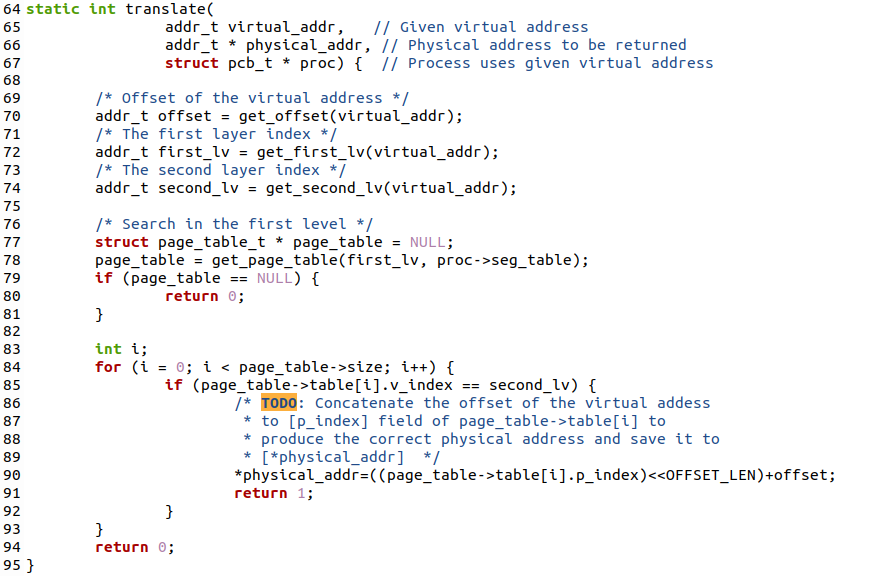
Hàm get\_page\_table ():



Nhiệm vụ là: thực hiện với index cho trước là i, đi qua mỗi hàng của segment table [seg\_table] và kiểm tra nếu v\_index của hàng đó bằng với index.

Ở dòng 55: dùng lệnh if nếu thõa thì trả về seg\_table->table[i].pages, thực hiện đến hết vòng lặp nếu đều không thõa thì trả về NULL.

Hàm translate ():



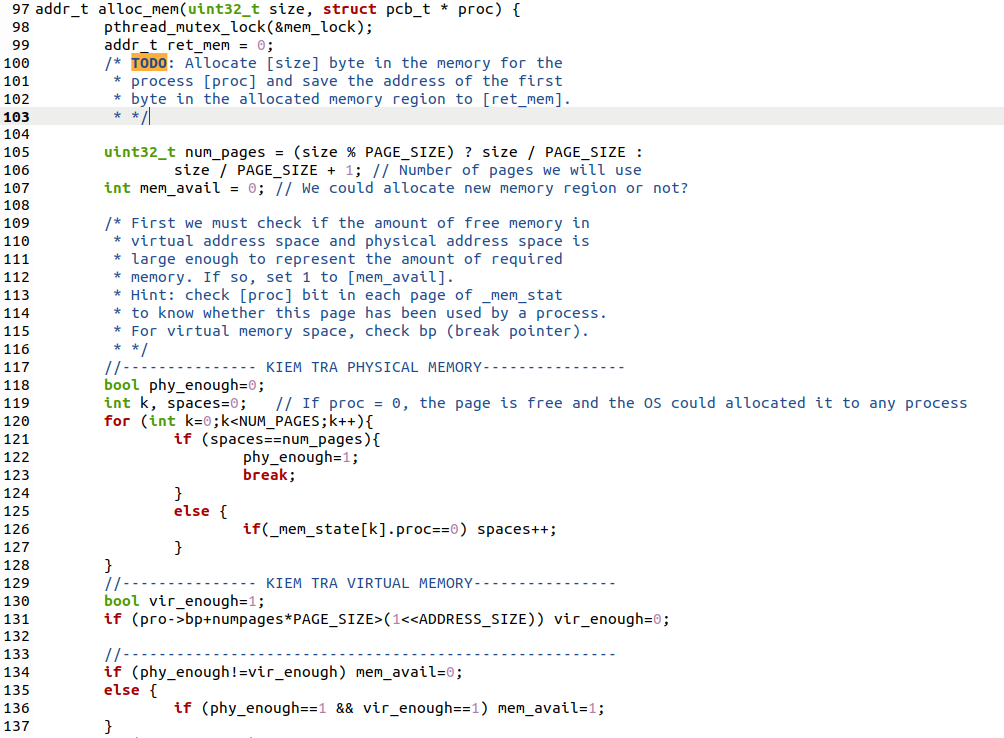
Nhiệm vụ được cho dòng (86-89): Ghép offset của địa chỉ ảo với [p\_index] của

page\_table->table[i] để tạo ra địa chỉ thực và lưu vào [\*physical\_addr]

Ở dòng 90: thực hiện nối bằng cách dịch trái [p\_index] khoảng OFFSET\_LEN sau đó cộng với offset của địa chỉ ảo.

Hàm alloc\_mem ():

Dùng mutex\_lock để tránh gây ra race condition



Nhiệm vụ được cho dòng (100-102): phân bổ [size[ byte trong bộ nhớ cho process [proc] và lưu địa chỉ byte đầu tiên trong vùng nhớ được phân bổ cho [ret\_mem]

Như gợi ý ở dòng (109-115) ta sẽ kiểm tra lượng memory free ở trong Physical Memory và Virtual Memory. Cụ thể ở Physical Memory ta sẽ kiểm tra trang nào free, nếu có thì tăng space lên đến khi spaces=num\_pages (số trang cần thiết) thì dừng lại. Sau đó kiểm tra Virtial Memory bằng cách check break pointer.

Break pointer có thể hiểu là địa chỉ bộ nhớ ảo cao nhất mà Hệ điều hành Linux đã cấp phát cho một process. Vì vậy ta sẽ kiểm tra bằng cách lấy từ break pointer cộng thêm lượng memory cần nếu vượt qua giới hạn của address size thì bộ nhớ ảo sẽ không đủ để đáp ứng.

Tổng hợp cả hai biến bool phy\_enough (check physical memory) và vir\_enough (check virtual memory) để xác định giá trị mem\_avail.



Sau khi kiểm tra nếu mem\_avail=1 thì ta thực hiện tiếp

Nhiệm vụ được cho ở dòng (140-145): thực hiện cập nhật trạng thái (status) của physical pages được phân bổ (allocated) cho [proc] trong \_mem\_stat, bao gồm:

* Cập nhật [proc], [index], [next]
* Thêm các entries vào segment table page tables of [proc] để đảm bảo rằng việc truy cập vào các vùng nhớ đã được phân bổ là có hiệu lực.

Cách làm:

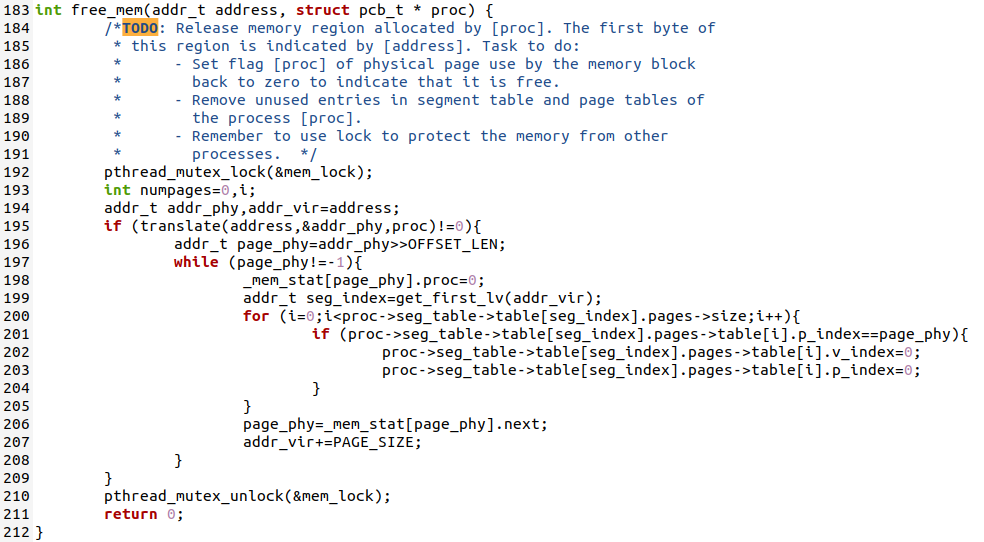
Ta sẽ đưa spaces=0, thực hiện update từng page, sau đó tăng dần spaces lên đến khi spaces=num\_pages thì dừng lại.

Đặt các biến seg\_index, page\_size, pre\_frame bằng -1.

Chạy vòng lặp và kiểm tra nếu \_mem\_stat=0 thì thực hiện update như sau:

* Cập nhật \_mem\_stat [index], [pid]
* Biến pre\_frame dùng để lưu index của frame trước đó, đặt giá trị ban đầu bằng -1, nếu thực hiện update trang đầu tiên thì bỏ qua pre\_frame sau đó cập nhật lại pre\_frame cho trang tiếp theo.
* Dùng hàm get\_first\_lv () để lấy seg\_index.
* Kiểm tra nếu page table là NULL thì cấp phát vùng nhớ, cập nhật size=0.
* Cập nhật giá trị page\_size, v\_index của seg\_table và v\_index, p\_index của page table.

Hàm free\_mem():



Nhiệm vụ được cho dòng (184-191): Giải phóng vùng nhớ được cấp phát cho [proc].   
Byte đầu tiên của vùng được biểu thị qua [address], thực hiện:

* Đặt flag [proc] của physical page sử dụng bởi memory block về giá trị 0 để biểu thị là nó free.
* Xóa các entries trong segment table và page tables của process [proc].
* Sử dụng lock để bảo vệ memory.

Cách làm:

Gán address vào addr\_vir. Kiểm tra nếu dịch địa chỉ ảo sang địa chỉ thực là hợp lệ, chạy vòng lặp và kiểm tra các physical page đã sử dụng sau đó thực hiện release và sang trang kế tiếp