ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

----------o0o----------

BÁO CÁO ĐỒ ÁN HỆ ĐIỀU HÀNH

Đồ án 2 Tìm hiểu syscalls - các thao tác với files

Nhóm 4

Giảng viên lý thuyết: Nguyễn Văn Giang.

Giảng viên thực hành: Lê Quốc Hòa.

MỤC LỤC

PHẦN 1. Thông tin thành viên

Nguyễn Phú Trí Nhân	Bùi Hoàng Vũ	Đặng Ngọc Tiến
20127580	20127668	20127641

PHẦN 2. Phiên bản Nachos đã sử dụng

- Phiên bản Nachos - OS 4.0
- Lập trình trên phiên bản Ubuntu 18.04

PHẦN 3. Mục tiêu đồ án và mức độ hoàn thành

1. *Mục tiêu:*

Hiểu và thực hiện các syscalls thao tác với files. Xây dựng các chương trình để vận dụng các syscall đã tạo.

2. *Mức độ hoàn thành:*

- Hoàn thành đầy đủ các syscall và xây dựng đầy đủ các chương trình mà đồ án yêu cầu.
- Xử lí các trường hợp ngoại lệ trong các syscall đầy đủ, không để người dùng làm sập hệ thống.
- Mỗi hàm trong các syscall được chú thích đầy đủ và rõ ràng.

PHẦN 4. Các file đã sử dụng:

- Trong quá trình xây dựng các syscall, chúng em đã xây dựng và bổ sung 1 số hàm hoặc file cần thiết để có thể hoàn thành được đồ án.

File của chương trình (có sẵn)	*File tự tạo*
ksycall.h	OpenFTable.h
exception.cc	
synconsole.h	
synconsole.cc	
filesys.h	
openfile.h	

openfile.cc	

- Trong file OpenFTable.h thì mục đích chúng em tạo file này để tạo một mảng con trỏ có dạng là OpenFile *. Việc này giúp chúng ta quản lý những file đã mở hoặc đóng. Trong file còn bao gồm các hàm như Openf, Readf, … để hỗ trợ cho các việc đọc và ghi file cùng nhiều những chức năng để thỏa mãn điều kiện của đồ án.

PHẦN 5. Phân tích các syscall đã xây dựng

1. Syscall OpenFile:

- Trong syscall này chúng em đã xây dựng một syscall để phục vụ cho việc mở một file.
- Tham số của syscall là một tên file muốn mở, được lấy thông qua user.
- Cách thực hiện:
  - Lấy tên tile từ người dùng.
  - Kiểm tra trong mảng OpenFile* trong file OpenFTable.h xem đã mở hay chưa, nếu đã mở sẽ trả về OpenFileId, điều này giúp ta hạn chế được thời gian tìm kiếm.
  - Nếu ở bước 2 mà không tìm thấy file ở trong bảng Table thì sẽ gọi hàm InsertF trong file OpenFTable.h, mục đích của việc này là thực hiện hai nhiệm vụ:
    - Nhiệm vụ tìm vị trí còn trống trong mảng OpenFile *, nếu có vị trí trống thì thực hiện bước tiếp theo, mục đích của việc làm này là để tìm ra 1 vị trí còn trống để chứa file cần mở. Nếu không có vị trí trống thì chúng ta trả về -1. Nếu có thì thực hiện bước tiếp theo.
    - Tiếp theo chúng ta sẽ gọi đến 1 hàm của hệ thống có sẵn là *OpenForReadWrite*, thực chất chúng ta có hai cách thực hiện là

OpenForRead hoặc OpenForReadWrite nhưng vì đề không yêu cầu nên chúng em mặc nhiên sử dụng hàm *OpenForReadWrite* cho tiện viết và đọc file hơn.

- Sau khi thực hiện hàm trên thì hàm sẽ trả về số sector trong Disk, chúng em sẽ dùng nó để tạo một con trỏ OpenFile * với private sector là giá trị trả về rồi trả về vị trí của file trong mảng.
- Ngược lại với trên, nếu khi số sector trả về là – 1, điều này mặc nhiên là do file không tồn tại nên chúng em cũng sẽ trả về -1.

- Để hiểu hơn về giá trị của hàm thì chúng em sẽ có bảng định nghĩa về giá trị trả về của syscall này như sau:

Giá trị trả về	*Ý nghĩa*
-1	File không tồn tại hoặc đã hết vị trí trong bảng Table.
Khác -1 và lớn hơn bằng 0	Tạo file thành công với giá trị trả về là index trong bảng OpenFile* hay còn gọi nó là OpenFileID.

- Một điểm lưu ý là ở bước tìm kiếm file đã tồn tại trong bảng OpenFTable hay chưa thì chúng em có 1 vài bổ sung như sau:
  - Vì trong class OpenFile.h các con trỏ trong mảng OpenFileTable chỉ lưu giá trị là sector nên khi chúng ta muốn so sánh file đã tồn tại hay không mà chỉ cung cấp tên file thì không thể nào xác định được điều này.
  - Để khắc phục hạn chế trên thì chúng em đã thêm 1 dữ liệu trong file *OpenFile.h là char* name*, mục đích là chứa tên file. Và để tiện cho việc sử dụng thì chúng em để phần dữ liệu này ở public.

2. *Syscall CreateFile:*

- Trong syscall này chúng ta sẽ xây dựng 1 syscall phục vụ cho việc tạo ra một file.
- Tham số của syscall này chính là 1 tên file bao gồm định dạng.
- Cách thực hiện:
  - Lấy tên file từ người dùng.
  - Kiểm tra tên file (độ dài và bộ nhớ). Nếu không thỏa mãn sẽ trả về -1.
  - Kiểm tra file có tồn tại hay chưa dựa vào một hàm mặc định *Open* nằm trong *filesystem.* Nếu file đã tồn tại thì sẽ hàm này sẽ trả về 1 con trỏ OpenFile * ngược lại sẽ trả về NULL. Chúng ta sẽ trả về -2 tương ứng khi file đã tồn tại.
  - Nếu file không tồn tại chúng ta sẽ gọi hàm *Create* của hệ thống trong *filesystem* để tạo file. Hàm này sẽ trả về False nếu tạo file thất bại và True nếu thành công.
- Ý nghĩa giá trị trả về:

Giá trị trả về	*Ý nghĩa*
-1	Tạo file thất bại hoặc lỗi liên quan đến vùng nhớ của tên file.
-2	Tạo file thất bại do file đã tồn tại.
0	Tạo file thành công

3. *Syscall CloseFile:*

- Trong syscall này chúng ta tạo ra một syscall hỗ trợ cho việc đóng một file đang mở.
- Tham số của syscall là *OpenFileID*, cụ thể hơn là vị trí của file trong bảng *Table*.
- Chúng ta sẽ kiểm tra giá trị đầu vào, dĩ nhiên giá trị này phải lớn hơn 1, vì sao lớn hơn 1 thì chúng em sẽ giải thích kĩ hơn ở phần sau. Và OpenfileID

phải nhỏ hơn số lượng phần tử tối đa của mảng. Ở đây thì người viết có thể tự cho giá trị khác nhưng nhóm của chúng em thì cho là tối đa là 20.

- Sau khi kiểm tra điều kiện của đầu vào chúng em sẽ kiểm tra tại vị trí đó, nếu con trỏ tại vị trí đó không bằng *NULL* thì chúng em sẽ giải phóng vùng nhớ tại đó cho và gán là *NULL* đồng thời trả về 0 nếu xóa thành công.
- Ý nghĩa giá trị trả về:

Giá trị trả về	*Ý nghĩa*
-1	Đóng file không thành công, do *OpenFileID* không hợp lệ hoặc là do file không tồn tại trong bảng *Table*.
0	Đã đóng file thành công.

4. *Syscall ReadFile:*

- Trong syscall này chúng ta sẽ hỗ trợ việc đọc một file với tên được xác định và đọc nó vào buffer, số lượng kí tự đọc được cho trước.
- Giải thích cho việc *OpenFileID* không được *nhỏ hơn 1* là bởi vì hai ô này trong Table sẽ dành cho *ConsoleIn(0)* và *ConsoleOut(1)* nên hai vị trí này phải được để trống. (Mục đích là để tương tác với *Console*).
- Tham số của syscall này bao gồm 3 tham số: buffer (dùng để chứa nội dung đọc được từ file), size (dùng để xác định bao nhiêu kí tự có thể được ghi vào buffer), id (dùng để xác định vị trí của file cần đọc trong bảng *Table*).
- Cách thực hiện:
  - Kiểm tra id của file có thỏa mãn điều kiện ràng buộc không (1 <= id < 20). Nếu id của file là 0 thì thực hiện lấy nội dung từ console vào buffer.

- Kiểm tra tại vị *Table[id]* có phải là *NULL* hay không, nếu phải thì trả về -1.
- Dựa vào một hàm có sẵn *Read (buffer, size)* đơn giản là đọc nội dung vào buffer.
- Trả về giá trị tương ứng.
- Ý nghĩa giá trị trả về:

Giá trị trả về	*Ý nghĩa*
-1	Đọc file thất bại, file không tồn tại hoặc chưa mở file. Có thể là lỗi buffer.
Số tự nhiên x lớn hơn 0	Trả về số byte đã đọc được.

5. *Syscall Writefile:*

- Trong syscall này chúng ta sẽ hỗ trợ việc viết vào một file với một bộ đệm và file cho trước.
- Tham số của syscall này tương tư như syscall *ReadFile* trước đó chỉ khác nhau là buffer này sẽ chứa dữ liệu để ghi vào file với id cho trước.
- Cách kiểm tra và thực hiện tương tự như ReadFile nhưng thay vì dùng hàm *Read* của hệ thống thì chúng ta dùng hàm *Write.*
- Bảng ý nghĩa giá trị trả về:

Giá trị trả về	*Ý nghĩa*
-1	Ghi file thất bại, file không tồn tại hoặc chưa mở file. Có thể là lỗi buffer.
Số tự nhiên x lớn hơn 0	Trả về số byte đã ghi được.

6. *Syscall Seek:*

- Trong syscall này chúng ta sẽ hỗ trợ việc di chuyển con trỏ trong file tới 1 vị trí bất kì trong file.

- Tham số của hàm này sẽ là *position* để cho biết vị trí muốn con trỏ di chuyển tới tròn file, *id* để cho biết vị trí của file trong mảng *Table*.
- Cách thực hiện:
  - Kiểm tra id, nếu nhỏ hơn 1 hoặc lớn hơn 20 thì sẽ trả về -1.
  - Kiểm tra *Table[id]* nếu là NULL thì sẽ trả -1.
  - Kiểm tra *position* nếu lớn hơn chiều dài của file thì sẽ trả về -2.
  - Nếu không vị phạm các điều kiện trên thì sẽ di chuyển con trỏ tới vị trí mong muốn và trả về *position* đã di chuyển. Đặc biệt nếu *position* thì sẽ mặc định di chuyển con trỏ tới cuối file.
- Bảng ý nghĩa giá trị trả về:

Giá trị trả về	***Ý nghĩa trả về***
-1	Position hoặc id không thỏa mãn.
-2	Giá trị của position lớn hơn chiều dài của file.
0	Di chuyển con trỏ thành công.

7. *Syscall Remove:*

- Trong syscall này chúng ta sẽ hỗ trợ xóa một file đã được tạo.
- Tham số của syscall này là một buffer chứa tên file cần xóa.
- Cách thực hiện:
  - Kiểm tra file có đang mở hay không, nếu có trả về -1.
  - Sử dụng hàm ***Remove*** sẵn có trong filesystem, nếu xóa thành công trả về 1, ngược lại trả về 0.
- Bảng ý nghĩa giá trị trả về:

Giá trị trả về	***Ý nghĩa trả về***
-1	File đang mở, không thể xóa.
0	Xóa file không thành công, có thể do file không tồn tại.
1	Xóa file thành công.

PHẦN 6. Các chương trình đã xây dựng

1. *Tổng quát chương trình xây dựng*

Tên chương trình	*Nội dung*
CreateFile.c	Tạo một file.
CopyFile.c	Copy 1 file từ file khác.
Cat.c	Đọc nội dung từ 1 file.
Delete.c	Xóa một file.
Concatenate.c	Nối nội dung hai file vào một file.

2. *Cách sử dụng từng file:*

a. *CreateFile.c:*

- Bước đầu tiên chúng ta phải nhập vào độ dài tên file và tên file.
  - Ví dụ:
    - 8
    - test.txt.
- Sau khi nhập vào thì sẽ nhận được thông báo tương ứng. Có thể thành công sẽ in ra thông báo ***"Successful"*** hoặc thất bại sẽ in ra lỗi tương ứng.
- Hình ảnh minh họa:
  - Trường hợp thành công:

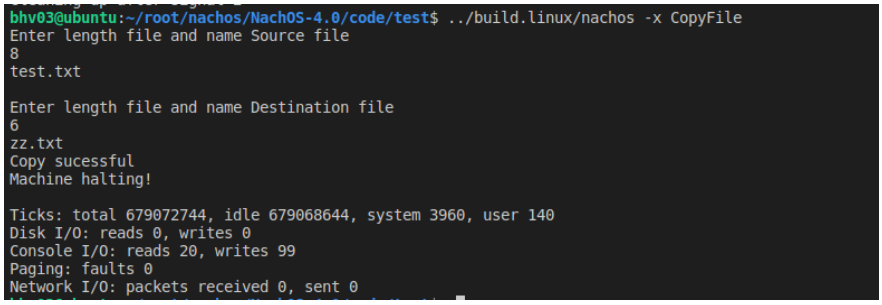
```
bhv03@ubuntu:~/root/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x CreateFile
Successful
Machine halting!

Ticks: total 1519, idle 1090, system 390, user 39
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 11
Paging: faults 0
Network I/O: packets received 0, sent 0
```

  - Trường hợp thất bại:

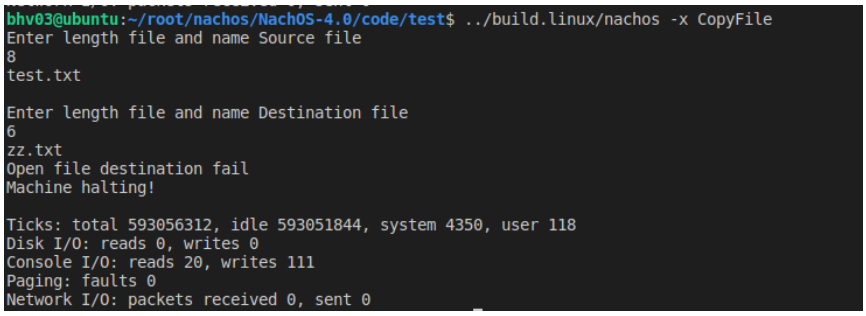
```
bhv03@ubuntu:~/root/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x CreateFile
File is exist
Machine halting!

Ticks: total 1919, idle 1390, system 490, user 39
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 14
Paging: faults 0
Network I/O: packets received 0, sent 0
```

b. *CopyFile.c:*

- Bước đầu tiên nhập độ dài và tên file chứa nội dung muốn sao chép.
- Tiếp theo, nhập vào độ dài và tên file chứa nội dung muốn dán vào.
- Sau khi nhập chúng ta sẽ có thông báo, nếu thành công sẽ thông báo *"Copy Success"*, ngược lại sẽ in ra lỗi tương ứng.
- Ví dụ minh họa:
  - Trường hợp thành công:



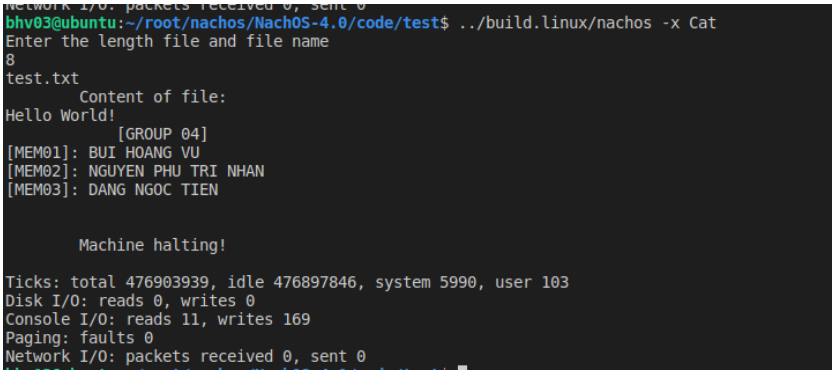
  - Trường hợp thất bại:



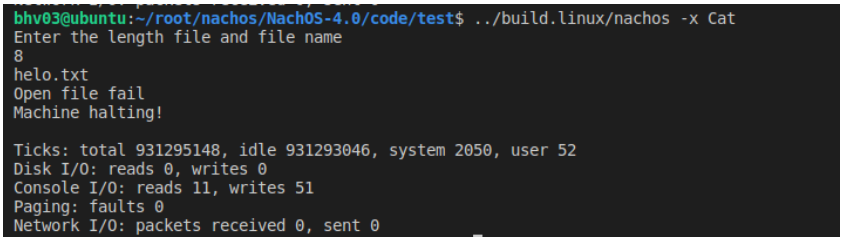
c. *Cat.c:*

- Nhập vào độ dài file và tên file có nội dung muốn xuất ra màn hình.
- Nếu thành công sẽ in ra nội dung của file, ngược lại sẽ in ra lỗi tương ứng.
- Ví dụ minh họa:

- Trường hợp thành công:



- Trường hợp thất bại:

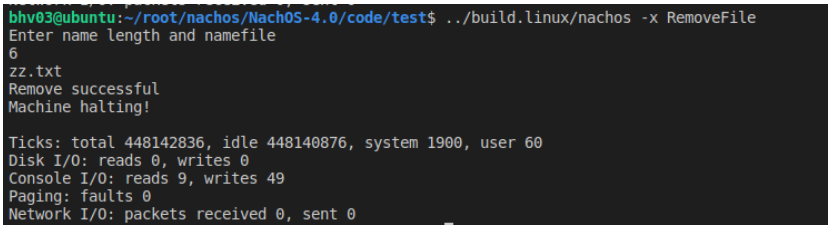


d. Concatenate.c

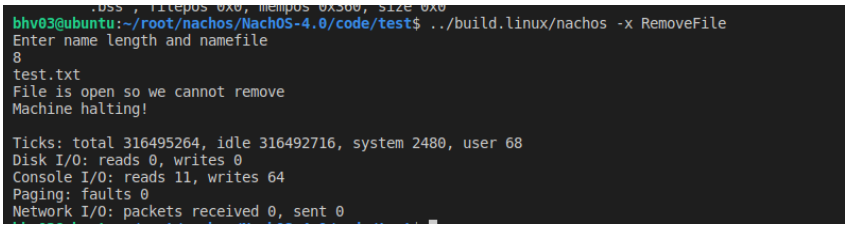
Tương tự như ***Copy.c***

e. RemoveFile.c

- Nhập vào độ dài và tên file muốn xóa.
- Nếu xóa thành công sẽ in ra "Remove Successful" ngược lại in ra lỗi tương ứng.
- Ví dụ minh họa:
  - Trường hợp thành công:



- Trường hợp thất bại:



PHẦN 7. Tổng kết

1. *Lời cảm ơn*

- Nhóm chúng em cảm ơn sự hướng dẫn từ hai Thầy trong lí thuyết và thực hành, đây là một môn học rất thú vị mà nhóm chúng em đã được thêm vào hành trang của mình.

2. *Tổng kết*

- Sau khi hoàn thành xong đồ án thì nhóm 4 chúng em đã hiểu hơn về việc thao tác với file đối với hệ điều hành.
- Trong quá trình thực hiện đồ án, chúng em không ít lần vấp phải những khó khăn trong việc quản lý bộ nhớ và phải liên kết nhiều file của hệ thống để tạo nên syscall hoàn chỉnh.
- Việc tạo thêm file sẽ đối diện với nhiều warning của chương trình, mặc dù file vẫn chạy được theo yêu cầu và chúng em đã cố tích cực chỉnh sửa code của mình nhưng đồ án vẫn chứa warning thì đây là một trong những hạn chế đồ án của nhóm chúng em.
- Mặc dù đã hoàn thành đồ án 100% theo yêu cầu, nhưng trong lúc dịch bệnh nên việc hợp tác của chúng em không thể không có thiếu sót, mong các Thầy thông cảm cho nhóm chúng em.

PHẦN 8. Tài liệu tham khảo

Mục đích	*Link / Tên tài liệu*
Hiểu rõ hơn về cách quản lý file của hệ điều hành. Đặc biệt hỗ trợ rất nhiều trong việc tạo file OpenFTable.h	Sách Operating System Concepts (Tenth Edition) – Part Six (File System)
Hiểu rõ hơn về các chức năng trong filesystem, openfile.h,directory.h và fileheader.h	Nachos Filesystem (duke.edu)
Video tham khảo	(90) [Nachos_01] Cài đặt nachos - YouTube