VNU - University of Science

Faculty of Information Technology

BÁO CÁO ĐỒ ÁN

Đồ án 1 : NachOS - Syscall for File System

Instructors: Phạm Tuấn Sơn

Lê Viết Long

Group Members: Võ Phạm Thanh Phương - 21127677

Nguyễn Quốc Huy - 21127511

Lê Hoàng Sang - 21127158

Hệ điều hành

Second Semester - 2022 - 2023

Table of content

1 Thông tin thành viên

MSSV	Họ tên	Đóng góp
21127677	Võ Phạm Thanh Phương	100%
21127511	Nguyễn Quốc Huy	100%
21127158	Lê Hoàng Sang	100%

2 Những thứ cần chuẩn bị trước khi cài đặt System call và exception

2.1 Hàm copy giữa 2 vùng nhớ

- Vào nachos/nachos-3.4/code/userprog, cụ thể là file exception.cc thực hiện khai báo hàm **char* User2System(int virtAddr,int limit)** và hàm **int System2User(int virtAddr,int len,char* buffer)** nhằm mục đích là thực hiện sao chép vùng nhớ giữa User space và System space.

2.2 Hàm tăng Program Counter

- Hàm **void IncreasePC()** trong file exception.cc dùng để tăng PC thêm 4 bytes lên sau mỗi lần gọi syscall thành công. Biến Previous PC sẽ được gán giá trị của PC hiện tại. Biến PC hiện tại sẽ được gán bằng Next PC. Và Next PC tăng thêm 4 bytes.
- Hàm có gọi đến 2 hàm **WriteRegister** và **ReadRegister** của con trỏ **machine** để ghi và đọc giá trị Program Counter.

2.3 Thông tin cơ bản về các thanh ghi

- Thanh ghi R1 và R3 do hệ thống sử dụng.
- Mã syscall và kết quả trả về của mỗi syscall sẽ được đưa vào thanh ghi R2.
- Tham số thứ nhất sẽ được đưa vào thanh ghi R4.
- Tham số thứ hai sẽ được đưa vào thanh ghi R5.
- Tham số thứ ba sẽ được đưa vào thanh ghi R6.

- Tham số thứ tư sẽ được đưa vào thanh ghi R7.

2.4 Các bước cài đặt system call

- Bước 1: Trong tập tin /code/userprog/syscall.h thêm dòng khai báo syscall mới
  Define giá trị cho syscall Create. Hàm Create trả về 0 nếu thành công và -1 nếu có lỗi. Và tương tự với các hàm Scanf, OpenFile, Seek,...

- Bước 2: Trong tập tin /code/test/start.c và /code/test/start.s thêm dòng khai báo hàm define các hàm Create, Seek,... để khai báo các hàm trong MIPS.

- Bước 3: Trong tập tin code/userprog/exception.cc sửa điều kiện **if** thành **switch case** (không cần thiết nhưng sẽ giúp code clean hơn) sao cho đủ các trường hợp lỗi và syscall. Nếu có lỗi thì in ra tên lỗi và Break. Nếu rơi vào syscallException thì chương trình tiếp tục check biến type và xác định gọi syscall nào từ class A ở file SCHandle.h. Sau mỗi lần gọi syscall thì biến PC sẽ được tăng lên để chương trình không bị loop.

- Bước 4: Viết chương trình ở mức người dùng trong thư mục test để kiểm tra và có sử dụng hàm đã khai báo ở trong code/userprog/syscall.h

- Bước 5: Thêm đoạn vào **Makefile** trong /code/test/
  Thêm tên hàm vào dòng all:

```
all: halt shell matmult sort <name>
```

  Ý nghĩa đoạn phía sau matmult: Ví dụ với hàm halt
  - Compile file halt.c thành file halt.o
  - Link file halt.o và start.o, sau đó tạo file thực thi halt.
  - Tạo file halt.coff (Linux) từ start.o và halt.o
  - Sử dụng coff2noff trong thư mục Bin để chuyển thành file halt.noff (NachOS).

- Bước 6: Biên dịch lại Nachos, cd tới .../nachos/code viết lệnh "make"

- Bước 7: Thực thi chương trình
  ./userprog/nachos -rs 1023 -x ./test/<tên file>

3 Lớp SynchConsole, OpenFile và FileSystem

3.1 Lớp SynchConsole

- Giúp truy xuất giữa console và người dùng và được cài đặt trong thư mục **code/threads**. Trong thư mục **threads/system.h**, phần định nghĩa macro của USER_PROGRAM, ta include header file của SynchConsole. Đồng thời ta khai báo thêm extern cho biến gSynchConsole

```
$ifdef USER_PROGRAM
#include "synchcons.h"
extern SynchConsole *gSynchConsole;
#endif
```

3.2 Lớp OpenFile

- Thêm một constructor cho OpenFile với tham số truyền vào là dạng file (**int mode**). Ngoài ra, thêm một attribute là biến **int seekPosition** để hỗ trợ system call Seek trong việc trỏ tới vị trí nào đó của file và một biến **type** để hỗ trợ nhận biết dạng file nào đang mở (0 hoặc 1)

3.3 Lớp FileSystem

- Thêm một biến con trỏ hai chiều **OpenFile **_openFile** là một table để lưu số lượng file đang mở trong hệ thống file và một biến **int pos** để lưu index của file trong table.
- Cài đặt lại constructor cho lớp FileSystem để cấp phát lại file table và reset **pos** về lại 0 và cài đặt destructor cho lớp FileSystem để tự động delete file table mỗi khi chương trình kết thúc để không bị leak memory

4 Cài đặt các hàm System call

Tổ chức các xử lí system call theo mô hình hướng đối tượng trong file SCHandle.h. Trong file SCHandle.h, ta tạo ra class SC_Handle với các hàm public tương ứng các hàm system call theo yêu cầu đề.

4.1 Syscall Halt()

- In ra thông báo shutdown.
- Thực hiện tạm dừng chương trình.

4.2 int CreateFile(char *name)

- Chức năng của **SC_CreateFile**:
  Input: Địa chỉ tên file ở User space
  Output: -1 nếu lỗi và 0 nếu thành công
  Mục đích: tạo một file rỗng với argument là tên file
- Đầu tiên, ta lấy tham số tên tập tin từ thanh ghi **r4**, sau đó sao chép giá trị lưu ở **r4** từ User space sang System space bằng hàm **User2System()**. Sau đó ta thực hiện kiểm tra coi tên file có phải NULL hay không. Nếu NULL thì return và ghi vào thanh ghi R2 giá trị -1. Nếu không NULL tức tạo thành công thì gọi Create từ fileSystem, trường hợp tạo không thành công thì ghi vào thanh ghi R2 giá trị -1. Nếu ghi thành công thì ghi vào R2 giá trị 0. Thực hiện xóa chuỗi filename để tránh bị memory leak.

4.3 OpenFileID Open(char *name, int type) và int Close(OpenFileID id)

4.3.1 OpenFileID Open(char *name, int type)

- Chức năng của **SC_Open**:
  Input: Địa chỉ của tên file ở User, chế độ mở
  Output: trả về Id của file nếu thành công, -1 nếu lỗi
  Mục đích: thực hiện mở file với tham số truyền vào là tên và chế độ mở
- Đầu tiên, ta đọc địa chỉ tên file từ thanh ghi **r4** và tham số type từ thanh ghi **r5**, sau đó ta kiểm tra xem vị trí của file trong fileSystem có nằm trong bảng mô tả file có kích thước là 10 file không (0<=pos<=9). Nếu không thì trả về -1.
- Tiếp theo, ta sao chép địa chỉ tên file từ User space sang System space bằng hàm **User2System()** và tiếp tục kiểm tra xem nếu tên file trùng với "stdin" thì trả về thanh ghi **r2** id của file là 0 hoặc nếu trùng với "stdout" thì trả về thanh ghi **r2** id của file là 1. 0 và 1 ở đây là index trong bảng mô tả file.

- Còn nếu là file bình thường thì trả về đúng id của file là **fileSystem->pos - 1** (pos-1 do hàm Open trong filesys.cc sẽ tăng pos của bảng mô tả file ngay sau khi mở file đó).
- Trong trường hợp file không tồn tại hoặc xảy ra lỗi về sai các điều kiện trên thì trả về -1 vào thanh ghi **r2**

4.3.2 int Close(OpenFileID id)

- Chức năng của **SC_Close**
  Input: id của file
  Output: Trả về -1 nếu lỗi hoặc 0 nếu thành công
  Mục đích: Đóng file với tham số truyền vào là id của file
- Đầu tiên, đọc tham số id của file từ thanh ghi **r4**, sau đó so sánh id của file với số lượng file từ fileSystem, biến i là số lượng file trong bảng, thực chất i sẽ tăng lên 1 ngay sau khi mở file (cần trừ 1), nhưng khi gán i = fileSystem->pos ta không trừ 1 vì mảng mô tả file bắt đầu từ 0.
- Nếu số lượng file nhỏ hơn id của file thì ghi -1 vào thanh ghi **r2** vì lỗi sai logic và xuất ra thông báo lỗi và return.
- Nếu thành công thì thực hiện xóa đi dữ liệu trong bảng _openFile[ ][ ] tại vị trí no. Sau đó ghi giá trị 0 vô thanh ghi R2, thành công.

4.4 int Read(char *buffer, int charcount, OpenFileID id)

- Chức năng của **SC_Read**
  Input: buffer(char*), số kí tự cho phép, id của file
  Output: trả về -1 nếu lỗi, -2 nếu đến cuối file, trả về số byte thật sự nếu thành công
  Mục đích: đọc file với 3 tham số trên
- Đầu tiên, đọc tham số id của file từ thanh ghi **r4**, sau đó so sánh id của file với số lượng file từ fileSystem (từ 0 đến 0). Nếu vượt quá bảng mô tả file thì trả về lỗi, giá trị -1 ở thanh ghi R2.
- Kế tiếp, kiểm tra xem tại vị trí pos đó trong bảng mô tả file có tồn tại file hay không. Nếu con trỏ tại _openFile[openf_id] là NULL, chứng tỏ không tồn tại file tại đó, trả về lỗi.

- Copy chuỗi virtAddr từ User sang System với bộ đệm buffer dài charcount.
- Xét điều kiện stdin, vị trí đầu bảng mô tả file. Sz là số byte thực sự được đọc, được trả về từ hàm Read của synchConsole. Tiếp theo, chuyển lại buf từ System sang User. Cuối cùng là ghi số byte đọc được thực tế vào R2, thoát chương trình.
- Trường hợp stdout ra màn hình, làm tương tự trường hợp stdin.
- Với trường hợp đọc file thông thường khác stdin và stdout, biến before vị trí byte lúc vừa mở file lên (có thể hiểu ở đầu file). Thực hiện đọc file, sau khi đọc xong ta tạo biến after lưu vị trí byte hiện tại của file (vị trí sau khi đọc hết file.
- Hiệu của after và before là số byte đọc thực sự, ghi vào thanh ghi R2, ta cộng 1 với ý nghĩa lấy cả byte before đang đứng tại đó.
- Cuối cùng, trường hợp đọc file thông thường không thành công ta ghi giá trị -1 vào thanh ghi R2.

4.5 int Write(char *buffer, int charcount, OpenFileID id)

- Chức năng của **SC_Write**
  Input: buffer(char*), số kí tự cho phép, id của file
  Output: trả về -1 nếu lỗi, rả về số byte thật sự nếu ghi thành công

  Mục đích: ghi ra file với 3 tham số trên
- Đầu tiên, ta đọc 3 tham số từ 3 thanh ghi R4, R5, R6. Ta xét các trường hợp id của file cần viết lớn hơn vị trí hiện tại, khi id lỗi (bé hơn 0) và trường hợp id bằng 0, tức stdin thì báo lỗi.
- Nếu tại vị trí id trong bảng mô tả file NULL, do file không tồn tại thì báo lỗi.
- Nếu tại vị trí id trong bảng mô tả file đang tồn tại file chỉ đọc với type là 1 thì báo lỗi.
- Chuyển charcount buffer từ User sang System.
- Trường hợp id trong bảng mô tả file là 1, tức stdout (ghi ra console), ta thực hiện ghi từng kí tự của buffer vào console đến khi gặp kí tự kết thúc. Trả về số byte ghi được tronh thanh ghi R2.

- Trường hợp cuối cùng là ghi ra file thông thường, biến before lưu vị trí byte lúc chưa ghi. Nếu ghi file thành công thì biến after vị trí byte sau khi vừa ghi xong. Ta ghi vào thanh hiệu after với before, ta cộng 1 với ý nghĩa lấy cả byte before đang đứng tại đó.

4.6 int Seek(int pos, OpenFileID id)

- Chức năng của **SC_Seek**
  Input: vị trí pos cần chuyển con trỏ tới, và id file trong bảng mô tả file
  Output: trả về vị trí thực sự trong file nếu thành công và -1 nếu bị lỗi
  Mục đích: chuyển con trỏ tới vị trí pos của file vị trí id trong bảng mô tả file

- Đầu tiên, ta đọc 2 tham số từ thanh ghi R4 và R5

- Kiểm tra các trường hợp file đó là stdin, stdout hoặc con trỏ vị trí id trong bảng file không tồn tại file thì báo lỗi, ghi giá trị -1 lên thanh ghi r2.

- Nếu file không bị rơi vào các trường hợp trên ta tính kích thước file, lưu vào biến len. Nếu vị trí pos cần seek lớn kích thước file, điều này là vô lý, ta trả về lỗi ở thanh ghi R2.

- Nếu pos là -1 thì ta gán pos bằng len, nhằm đưa pos về vị trí cuối file theo yêu cầu đề.

- Cuối cùng ta đưa con trỏ của file trong bảng mô tả file đó về vị trí pos theo yều cầu. Và ghi giá trị pos vào thanh ghi R2.

4.7 int Delete(char *name, OpenFileID id)

- Chức năng của **SC_Delete**
  Input: địa chỉ mảng chứa tên tập tin, ta thêm vào id file để mở file trong bảng mô tả file
  Output: trả về 0 nếu xóa thành công và -1 nếu xóa không thành công
  Mục đích: xóa file trong bảng mô tả file

- Đầu tiên, ta đọc tham số địa chỉ tên tập tin và id file

- Chuyển tên file từ User space sang System space.

- Sau khi chuyển, trong trường hợp bộ nhớ System không còn đủ chỗ cho tên file, trả về lỗi.

- Nếu không, ta tiếp tục kiểm tra giới hạn hợp lệ của id trong bảng mô tả file.

- Tiếp theo, file đang tồn tại nhưng ta cần kiểm tra xem nó có đang mở không, nếu file không mở (tương ứng với bảng _openFile tại vị trí đó là NULL), thực hiện gọi hàm Remove của fileSystem. Nếu xóa thành công thì ghi 0 vào thanh ghi R2, ngược lại ghi R1 vào thanh ghi R2.
- Trong trường hợp file đó đang mở thì ta không thể xóa và trả về lỗi.

4.8 Syscall Print()

- Chức năng của **Syscall_Print()**
  Input: tham số cần in
  Output: void
  Mục đích: dùng lớp synchConsole để in ra nội dung tham số
- Đầu tiên, đọc tham số từ thanh ghi R4 và chuyển nó sang System space.
- Duyệt từng kí tự của mảng đó đến khi gặp kí tự kết thúc. Viết từng kí tự vào lớp synchConsole.

4.9 Syscall Scan()

- Chức năng của **SC_Scan**
  Input: địa chỉ buffer và độ dài
  Output: void
  Mục đích:
- Đầu tiên, đọc 2 tham số buffer và độ dài từ thanh ghi R4 và R5
- Gọi hàm đọc của lớp synchConsole, hàm trả về kích thước sz byte đọc được thật sự.
- Cuối cùng, chuyển sz byte đó về lại User space.

5 Viết chương trình Test

5.1 Chương trình createfile.c để kiểm tra system call CreateFile

- Yêu cầu người dùng được nhập vào từ console bằng hàm Scanf(filename, MAX_LENGTH)
- Sau đó dùng system call Create để tạo file, in ra thông báo thành công hoặc thất bại bằng hàm Printf sau khi tạo file.

- Demo

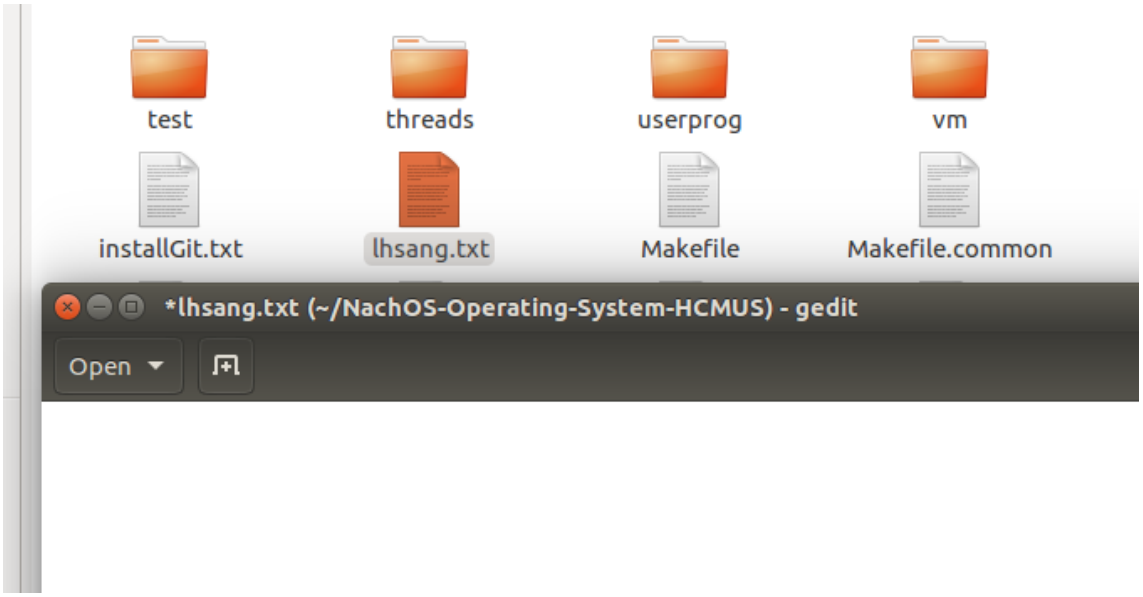
```
sang@ubuntu:~/NachOS-Operating-System-HCMUS$ ./userprog/nachos -rs 1023 -x ./test/createfile
Input file name to create: lhsang.txt

Successfully create file lhsang.txt

 Shutdown, initiated by user program.Machine halting!

Ticks: total 1021577393, idle 1021575033, system 2310, user 50
Disk I/O: reads 0, writes 0
Console I/O: reads 11, writes 63
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
Assertion failed: line 254, file "../machine/sysdep.cc"
Aborted (core dumped)
sang@ubuntu:~/NachOS-Operating-System-HCMUS$
```



5.2 Chương trình echo.c

- Chương trình này cho phép mỗi khi nhập dòng từ console thì console xuất lại dòng đó
- Đầu tiên, ta in ra yêu cầu nhập và gọi đến system call Scanf để đọc nội dung nhập từ stdin vào buffer, tiếp theo gọi đến system call Print để ghi nội dung vừa đọc được từ buffer ra stdout.
- Demo

```
Ticks: total 670026719, idle 670025492, system 1160, user 67
Disk I/O: reads 0, writes 0
Console I/O: reads 8, writes 30
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
thanhphuong@ubuntu:~/Desktop/HDH/nachos/nachos-3.4/code$ userprog/nachos -rs 102
3 -x test/echo
Enter a string: Vo Pham Thanh Phuong
The input string: Vo Pham Thanh Phuong
```

5.3 Chương trình cat.c

- Chương trình này yêu cầu nhập vô filename, rồi hiện thị nội dung của file đó.
- Đầu tiên, ta gọi đến system call Scanf để nhập vào filename. Sau đó thực hiện gọi đến system call Open với tham số truyền vào là filename vừa đọc được ở trên. Nếu không mở được thì thông báo lỗi và kết thúc chương trình.
- Nếu mở thành công thì gọi đến system call Seek để di chuyển đến cuối file và gọi lại lần hai di chuyển đến đầu file để lấy vị trí byte cuối và đầu file.
- Sau đó tiến hành gọi đến system call Read để đọc từng kí tự trong file và in từng kí tự ra console bằng system call Print. Cuối cùng đóng file bằng system call Close và dùng system call Halt để trả hệ thống về cho người dùng.
- Demo

```
sang@ubuntu:~/NachOS-Operating-System-HCMUS$ ./userprog/nachos -rs 1023 -x ./test/cat
Input file name:lhsang.txt

Open file success 'lhsang.txt'
huy, sang, phuong - hcmus

Read file done

Close file success


 Shutdown, initiated by user program.Machine halting!

Ticks: total 506937406, idle 506934412, system 2210, user 784
Disk I/O: reads 0, writes 0
Console I/O: reads 11, writes 58
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
Assertion failed: line 254, file "../machine/sysdep.cc"
Aborted (core dumped)
sang@ubuntu:~/NachOS-Operating-System-HCMUS$
```

```
sang@ubuntu:~/NachOS-Operating-System-HCMUS$ ./userprog/nachos -rs 1023 -x ./test/copy
Input source file: abc.txt

Input destination file: def.txt

Open file success 'abc.txt'

Open file success 'abc.txt'

Open file success 'abc.txt'

Open file success 'abc.txt'

Open file success 'def.txt'

Copy file successfully

Close file success

Close file success


 Shutdown, initiated by user program.Machine halting!

Ticks: total 946840163, idle 946836894, system 2540, user 729
Disk I/O: reads 0, writes 0
Console I/O: reads 16, writes 68
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
Assertion failed: line 254, file "../machine/sysdep.cc"
Aborted (core dumped)
```

5.4 Chương trình copy.c

- Chương trình này yêu nhập tên file nguồn và file đích và thực hiện copy nội dung.
- Đầu tiên, dùng system call Printf và Scanf để thông báo và cho người dùng nhập vào tên file nguồn và đích.
- thực hiện mở 2 file bằng Open và thông báo lỗi nếu có (dừng chương trình nếu có lỗi). Thông báo thành công nếu 2 file nguồn và đích được mở thành công.
- Trường hợp file nguồn mở thành công nhưng file đích mở không thành công, in màn hình lựa chọn cho phép người dùng nhập lại file đích hoặc thoát chương trình.
- Cuối cùng, nếu 2 file mở thành công thì ra tính kích thước file nguồn, và bắt đầu đọc file nguồn từ vị trí 0 và ghi sang file đích đến khi con trỏ đến vị trí bằng với số byte của file nguồn, lúc này ta đã copy file thành công. Thực hiện đóng 2 file và halt chương trình.
- Demo

```
sang@ubuntu:~/NachOS-Operating-System-HCMUS$ ./userprog/nachos -rs 1023 -x ./test/copy
Input source file: abc.txt

Input destination file: non.txt

Open file success 'abc.txt'

Open file success 'abc.txt'

Open file success 'abc.txt'

Open file success 'abc.txt'

File destination have trouble when open
Do you want to continue with a new file (Y/n): y

Please input a new file name: non.txt
Can not open file 'non.txt'
Open file success 'non.txt'

Copy file successfully

Close file success

Close file success


 Shutdown, initiated by user program.Machine halting!

Ticks: total 1805911945, idle 1805904327, system 6780, user 838
Disk I/O: reads 0, writes 0
Console I/O: reads 26, writes 187
Paging: faults 0
```

5.5 Chương trình delete.c

- Chương trình này dùng để xóa file
- Đầu tiên, cho người dùng nhập vào tên file. Thực hiện mở file, ta có được id của file đó, ta thử xóa file khi đang mở và kết quả trả về là không được. Tiếp đó ta đóng file và thực hiện delete lại, kết quả thành công.
- Demo

```
sang@ubuntu:~/NachOS-Operating-System-HCMUS$ ./userprog/nachos -rs 1023 -x ./test/delete
Input file name: lhsang.txt

Open file success 'lhsang.txt'
Opened file

 File 'lhsang.txt' is opening, can't be deleted
Close file success

Delete file 'lhsang.txt' successfully

 Shutdown, initiated by user program.Machine halting!

Ticks: total 464106209, idle 464104972, system 1170, user 67
Disk I/O: reads 0, writes 0
Console I/O: reads 11, writes 30
Paging: faults 0
Network I/O: packets received 0, sent 0

Cleaning up...
Assertion failed: line 254, file "../machine/sysdep.cc"
Aborted (core dumped)
sang@ubuntu:~/NachOS-Operating-System-HCMUS$
```