Đại học quốc gia thành phố Hồ Chí Minh

Trường Đại học Khoa Học Tự Nhiên

Hệ điều hành

Đồ án 1: Exceptions và các system calls đơn giản

Nhóm thực hiện:

Nguyễn Văn Phước	1612523
Châu Hoàng Phúc	1612520
Nguyễn Thị Thu Quyền	1612548
Nguyễn Trương Quang	1612533

Giáo viên:

Phạm Tuấn Sơn

MỤC LỤC

I. LỚP FILEMANAGE: 3

1. CHỨC NĂNG: 3
2. CÀI ĐẶT: 3
   a. *Cấu trúc FileCustom:* *3*
   b. *Cấu trúc FileManage:* *3*

II. CREATEFILE SYSCALL: 4

1. CHỨC NĂNG: 4
2. CÁCH CÀI ĐẶT: 4

III. OPEN SYSCALL: 5

1. CHỨC NĂNG: 5
2. CÁCH CÀI ĐẶT: 5

IV. CLOSEFILE SYSCALL: 6

1. CHỨC NĂNG: 6
2. CÁCH CÀI ĐẶT: 6

V. READ SYSCALL: 7

1. CHỨC NĂNG: 7
2. CÁCH CÀI ĐẶT: 7

VI. WRITE SYSCALL: 8

1. CHỨC NĂNG: 8
2. CÁCH CÀI ĐẶT: 8

VII. SEEK SYSCALL: 10

1. CHỨC NĂNG: 10
2. CÁCH CÀI ĐẶT: 10

I. Lớp FileManage:

1. Chức năng:

- Có nhiệm vụ quản lý tập tin, cung cấp các phương thức mở tập tin, đóng và seek.

2. Cài đặt:

a. Cấu trúc FileCustom:

- Có chức năng lưu trữ thông tin của tập tin, con trỏ OpenFile, vị trí đọc ghi file và loại tập tin.
- Có cấu trúc như sau:
  + Một con trỏ OpenFile để lưu thông tin mở file
  + Biến pos lưu vị trí đọc ghi
  + Biến type lưu loại tập tin

b. Cấu trúc FileManage:

- Sử dụng một con trỏ 2 chiều kiểu FileCustom để lưu thông tin tất cả các file được mở trong chương trình.
- Gồm các phương thức sau:

  + **FindFreeSlot** tìm vị trí còn trống để lưu thông tin file, trả về -1 nếu không còn vị trí trống, ngược lại trả về vị trí trống.

  + **Add** thêm thông tin file mới vào mảng thông tin file, trả về true nếu thêm thành công, ngược lại trả về false

  + **GetFile** lấy ra con trỏ FileCustom để xử lý đọc ghi dữ liệu, trả về con trỏ FileCustom tương ứng nếu tồn tại, ngược lại trả về null.

  + **Seek** dịch vị trí đọc ghi file, trả về -1 nếu có lỗi, ngược lại trả về vị trí dịch thật tế.

  + **CloseFile** đóng file đã mở, trả về true nếu đóng thành công, ngược lại trả về false.

II. CreateFile syscall:

1. Chức năng:

- Syscall hỗ trợ tạo mới tập tin, với tham số truyền vào là tên file.
- Trả về -1 nếu tạo file thất bại, và 0 nếu tạo file thành công

2. Cách cài đặt:

- Ta định nghĩa SC_Create và khai báo hàm vào tập tin **syscall.h**

```
int CreateFile (char *name)
```

- Ánh xạ syscall vào trong file **start.c** và **start.s**:

```
        .globl CreateFile
        .ent CreateFile
CreateFile:
        addiu $2, $0, SC_Create
        syscall
        j $31
        .end CreateFile
```

- Vào file **exception.cc** và xử lý với switch case có type bằng với SC_Create. Trong đây ta sẽ định nghĩa các công việc thực hiện của syscal như sau:
    - Đọc và sao chép tên file từ vùng userspace sang vùng kernel space thông qua hàm User2System
    - Kiểm tra xem tên file có NULL hay không, nếu NULL thì trả giá trị về là -1
    - Nếu tên file khác NULL thì ta sử dụng đối tượng **fileSystem** đã được khai báo ở **system.h** để tạo file mới thông qua hàm Create của fileSystem.
    - Kiểm tra giá trị trả về sau khi gọi hàm Create của filesystem, nếu trả về là **true** thì là tạo file thành công, ngược lại là tạo file thất bại.
    - Trả về giá trị cho user nếu tao thành công thì trả về 0 ngược lại trả về -1

III. Open syscall:

1. Chức năng:

- Mở tập tin với các tùy chọn: chỉ đọc hoặc đọc ghi
- Trả về id của file nếu mở file thành công ngược lại thì trả về -1

2. Cách cài đặt:

- Định nghĩa SC_Open và khai báo hàm trong **syscall.h**

```
OpenFileId Open (char *name, int type)
```

- Tiếp tục đinh nghĩa các kiểu mở file như sau:
  - ReadOnly mở file chỉ dùng để đọc, không cho phép ghi dữ liệu
  - ReadWrite mở file dùng để đọc và ghi.
- Ánh xạ syscall vào file **start.c** và **start.s**

```
        .globl Open
        .ent Open
Open:
        addiu $2, $0, SC_Open
        syscall
        j $31
        .end Open
```

- Cài đặt syscall trong file **exception.cc** với type bằng SC_Open:
  - Đọc tên file, chuyển sang vùng kernel và kiểm tra có NULL hay không, nếu NULL thì trả về -1
  - Đọc và kiểm tra xem kiểu mở file có hợp lệ hay không. Nếu không trả về -1.
  - Sau khi kiểm tra tên và kiểu mở file hợp lệ thì ta sử dụng hàm **Open** của fileSystem để mở file. Hàm Open của fileSystem sẽ trả về con trỏ OpenFile ứng với tập tin đó nếu mở file thành công, ngược lại trả về NULL.
  - Nếu trả về NULL thì ta đưa giá trị trả về của syscall là -1
  - Ngược lại ta tiến hành sử dụng đối tượng **fileManage** của lớp FileManage được khai báo ở lớp thread để lưu lại thông tin của file:

- Sử dụng phương thức **FindFreeSlot** để tìm vị trí lưu trữ đồng thời kiểm tra xem còn vị trí trống hay không. Nếu không thì báo lỗi và dừng syscall.
- Nếu còn vị trí trống thì sử dụng phương thức **Add** để thêm thông tin file vào vị trí đã tìm ở trên. Nếu lưu thành công thì sẽ trả về true, ngược lại trả về false, ta dựa vào đó để trả giá trị về cho người dùng.

IV. CloseFile syscall:

1. Chức năng:

- Đóng file đã mở.
- Trả về 0 nếu đóng thành công, -1 nếu file chưa được mở.

2. Cách cài đặt:

- Định nghĩa SC_Close và khai báo hàm trong syscall.h

```
int CloseFile (OpenFileId id)
```

- Ánh xạ vào file **start.c** và **start.s**:

```
        .globl CloseFile
        .ent CloseFile
CloseFile:
        addiu $2, $0, SC_Close
        syscall
        j $31
        .end CloseFile
```

- Cài đặt syscall trong **exception.cc**, với type bằng SC_Close:
  - Lấy id từ người dùng truyền vào và kiểm tra xem có hợp lệ hay không (phải lớn hơn 1 và nhỏ hơn 10, vì ta không thể đóng được ConsoleInput và ConsoleOutput, và giới hạn chỉ có 10 file), nếu không hợp lệ trả về -1.
  - Sau đó ta kiểm tra xem giá trị của FileCustom tại vị trí id trong mảng, nếu FileCustom bằng NULL thì trả về -1, vì file chưa được mở.

- Ngược lại thì ta sử dụng phương thức CloseFile của đối tượng fileManage để đóng file, nếu đóng thành công trả về true, ngược lại trả về false. Sau đó trả dữ liệu về cho người dùng.

V. Read syscall:

1. Chức năng:

- Đọc dữ liệu từ file hoặc console.
- Trả về số byte đọc được
- Nếu đọc file lỗi thì trả về -1, hoặc -2 nếu là cuối file.

2. Cách cài đặt:

- Định nghĩa SC_Read và khai báo hàm trong **syscall.h**

  int Read (char *buffer, int charcount, OpenFileId id)
- Ánh xạ vào **start.c** và **start.s:**

```
.globl Read
.ent Read
Read:
    addiu $2, $0, SC_Read
    syscall
    j $31
    .end Read
```

- Cài đặt syscall trong **exception.cc** với type bằng SC_Read:
  - Đọc địa chỉ vùng data.
  - Đọc số byte cần đọc, và kiểm tra nếu số byte cần đọc bằng 0 thì trả về 0, nếu số byte cần đọc nhỏ hơn 0 thì trả về -1.
  - Kiểm tra thông tin file mà có id truyền vào nếu id bằng với ConsoleInput (chỉ ghi) thì trả về -1.
  - Nếu id bằng với ConsoleOutput thì ta tiến hành đọc dữ liệu từ console thông qua đối tượng **gSynchConsole** của lớp **SynchConsole** do ta đã định nghĩa ở **system.h**, ta sử dụng hàm Read của gSynchConsole để đọc dữ liệu từ console vào con trỏ **buffer** và kiểm tra giá trị trả về nếu trả về là -1 thì đọc lỗi ta sẽ trả về cho người dùng -1, nếu trả về -2 thì đây là cuối console ta sẽ trả về -2, ngược lại ta sẽ tiến hành copy vùng nhớ của **buffer**

hiện đang ở vùng kernel sang vùng nhớ của user bằng hàm System2User và trả về cho người số byte đã được copy sang vùng userspace trả ra từ hàm System2User.

- Ngược lại ta tiến hành xử lý đọc dữ liệu với file:
  - Sử dụng phương thức **GetFile** của đối tượng fileManage để lấy ra đối tượng FileCustom ứng với file cần đọc, nếu trả về NULL thì trả về -1 và dừng syscall.
  - Kiểm tra xem vị trí đọc file hiện tại của file và dung lượng file, nếu vị trí mà lớn hơn hoặc bằng dung lượng file thì trả về -2 (cuối file).
  - Ngược lại ta tiến hành đọc file thông qua hàm ReadAt của đối tượng file trong FileCustom là con trỏ đối tượng OpenFile được lưu lại khi mở file. Hàm ReadAt cho phép ta đọc dữ liệu của file vào **buffer** với số lượng byte cho trước tại vị trí truyền vào.
  - Ta sẽ tiến hành đọc file tại vị trí mà file đang trỏ tới qua biến pos trong FileCustom, và kiểm tra giá trị trả về của hàm ReadAt, nếu trả về -1 (đọc lỗi) thì ta trả về -1 cho người dùng, ngược lại tiến hành copy **buffer** từ kernel space sang user space bằng System2Use.
  - Cập nhật lại pos của FileCustom và trả giá trị là số byte mà System2User copy được.

VI. Write syscall:

1. Chức năng:

- Ghi dữ liệu xuống file hoặc console
- Trả về số byte đã được ghi
- Nếu ghi file lỗi thì trả về -1.

2. Cách cài đặt:

- Định nghĩa SC_Write và khai báo hàm trong **syscall.h**

  int Write (char *buffer, int charcount, OpenFileId id)

- Ánh xạ vào **start.c** và **start.s:**

```
.globl Write
.ent Write
Write:
addiu $2, $0, SC_Write
syscall
j $31
.end Write
```

- Cài đặt syscall trong **exception.cc** với type bằng SC_Write:
  - Đọc địa chỉ vùng data.
  - Đọc số byte cần ghi, nếu số byte cần ghi bằng 0 thì trả về 0, nếu số byte cần ghi nhỏ hơn 0 thì trả về -1.
  - Kiểm tra thông tin file mà có id truyền vào nếu id bằng với ConsoleOutput (chỉ đọc) thì trả về -1.
  - Ngược lại ta tiến hành copy vùng data từ user space sang kernel space bằng hàm User2System, kiểm tra giá trị **buffer** trả về nếu bằng NULL thì trả về -1, ngược lại làm bước tiếp theo.
  - Nếu id bằng với ConsoleInput thì ta sử dụng hàm Write của đối tượng gSynchConsole và kiểm tra giá trị trả về, nếu trả về -1 thì ta trả về -1 cho người dùng, ngược lại thì trả về giá trị mà gSynchConsole ghi được.
  - Ngược lại ta xử lý ghi file:
    - Sử dụng phương thức **GetFile** của fileManage để lấy ra đối tượng FileCustom ứng với file cần ghi, nếu trả về NULL thì báo lỗi và kết thúc syscall.
    - Kiểm tra file có id truyền vào có cho phép quyền ghi hay không (ReadWrite mode), nếu không ta trả về -1.
    - Ngược lại ta sử dụng hàm WriteAt của đối tượng **file** trong FileCustom để tiến hành ghi file tại vị trí pos.
    - Nếu giá trị trả về của hàm WriteAt bằng -1 (ghi lỗi) thì ta trả về -1 cho người dùng, ngược lại ta cập nhật lại **pos** và trả về số byte đã được ghi.

VII. Seek Syscall:

1. Chức năng:

- Dịch chuyển vị trí đọc, ghi file tới vị trí mong muốn.
- Trả về -1 nếu lỗi, ngược lại trả về vị trí thực được dịch tới.

2. Cách cài đặt:

- Định nghĩa SC_Seek và khai báo hàm trong syscall.h

```
int Seek (int pos, OpenFileId id)
```

- Ánh xạ vào file start.c và start.s

```
        .globl Seek
        .ent Seek
Seek:
        addiu $2, $0, SC_Seek
        syscall
        j $31
        .end Seek
```

- Cài đặt syscall trong **exception.cc** với type bằng SC_Seek
    - Đọc id file và vị trí position seek tới truyền vào hàm **Seek** của đối tượng fileManage để tiến hành chuyển vị trí đọc ghi của file tới vị trí mới.
    - Kiểm tra giá trị trả về nếu trả về -1 là seek lỗi, ngược lại sẽ trả về vị trí seek tới. Lúc này ta sẽ trả về giá trị cho người dùng và thoát syscall.