**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

------------------🙦🕮🙤------------------

**Logo

Description automatically generated**

**BÁO CÁO ĐỒ ÁN 02**

**LẬP TRÌNH NACHOS**

**HỆ ĐIỀU HÀNH**

**Lớp 19\_21**

**GVHD: ThS. Lê Viết Long**

**Thành phố Hồ Chí Minh – 2021**

Mục lục

[**1.** **Bảng đánh giá thành viên** 3](#_Toc90150199)

[**2.** **Tổng quan về NachOS** 3](#_Toc90150200)

[2.1. Hiểu được mã nguồn của Nachos 3](#_Toc90150201)

[2.2. Hiều được thiết kế của chương trình Nachos 4](#_Toc90150202)

[**3.** **Cài đặt System call và exception** 9](#_Toc90150203)

[3.1. Hàm IncreasePC() 9](#_Toc90150204)

[3.2. System call ReadInt 9](#_Toc90150205)

[3.3. System call PrintInt 11](#_Toc90150206)

[3.4. System call ReadChar 13](#_Toc90150207)

[3.5. System call PrintChar 14](#_Toc90150208)

[3.6. System call ReadString 15](#_Toc90150209)

[3.7. System call PrintString 15](#_Toc90150210)

[**4.** **Demo minh họa** 16](#_Toc90150211)

[4.1. Chương trình help 17](#_Toc90150212)

[4.2. Chương trình in bảng ascii 17](#_Toc90150213)

[4.3. Chương trình sort (mysort) 18](#_Toc90150214)

[**5.** **Tài liệu tham khảo** 20](#_Toc90150215)

1. **Bảng đánh giá thành viên**

|  |  |  |  |
| --- | --- | --- | --- |
| MSSV | Họ Tên | Công việc được giao | Đóng góp |
| 19120532 | Nguyễn Thanh Huy |  | 100% |
| 19120666 | Nguyễn Hữu Thịnh |  | 100% |
| 19120667 | Nguyễn Văn Thịnh |  | 100% |
| 19120671 | Lê Nguyễn Nhất Thọ | Viết các chương trình demo + Kiểm thử chương trình + Tổng hợp báo cáo | 100% |
| 19120675 | Huỳnh Mạnh Tiến |  | 100% |

Mức độ hoàn thành đồ án **80%**

Chương trình có các comments, ngắn nhưng đầy đủ.

Thiết kế và cài đặt các system call.

Xây dựng các chương trình demo

1. **Tổng quan về NachOS**
   1. **Hiểu được mã nguồn của Nachos**

* Hệ thống NachOS được chia làm ba thành phần chính: Các chương trình ứng dụng (...code/test), Cỗ máy ảo MIPS, Hệ điều hành NachOS.
* **progtest.cc** kiểm tra các thủ tục để chạy chương trình người dùng
* **syscall.h** system call interface: các thủ tục ở kernel mà chương trình người dùng có thể gọi
* **exception.cc** xử lý system call và các exception khác ở mức user, ví dụ như lỗi trang, trong phần mã chúng tôi cung cấp, chỉ có ‘halt’ system call được viết
* **bitmap.\*** các hàm xử lý cho lớp bitmap (hữu ích cho việc lưu vết các ô nhớ vật lý) **filesys.h**
* **openfile.h** định nghĩa các hàm trong hệ thống file nachOS. Trong đồ án này chúng ta sử dụng lời gọi thao tác với file trực tiếp từ Linux, trong đồ án khác chúng ta sẽ triển khai hệ thống file trên ổ đĩa giả lập. (nếu kịp thời gian)
* **translate.\*** Phiên bản nachos chúng tôi gửi các bạn, chúng tôi giả sử mỗi địa chỉ ảo là cũng giống hệt như địa chỉ vật lý, điều này giới hạn chúng ta chỉ chạy 1 chương trình tại một thời điểm. Các bạn có thể viết lại phần này để cho phép nhiều chương trình chạy  cùng lúc trong đồ án sau.
* **machine.\*** mô phỏng các thành phần của máy tính khi thực thi chương trình người dùng: bộ nhớ chính, thanh ghi, v.v.
* **mipssim.cc** mô phỏng tập lệnh của MIPS R2/3000 processor
* **console.\*** mô phỏng thiết bị đầu cuối sử dụng UNIX files. Một thiết bị có đặc tính (i) đơn vị dữ liệu theo byte, (ii) đọc và ghi các bytes cùng một thời điểm, (iii) các bytes đến bất đồng bộ
* **synchconsole.\*** nhóm hàm cho việc quản l y’ nhập xuất I/O theo dòng trong Nachos.
* **../test/\*** Các chương trình C sẽ được biên dịch theo MIPS và chạy trong Nachos
  1. **Hiều được thiết kế của chương trình Nachos**

Nachos là một phần mềm mã nguồn mở giải lập một máy tính ảo và một số thành phần cơ bản của hệ điều hành trên máy tính ảo này nhằm giúp cho việc tìm hiểu và xây dựng các thành phần phức tạp khác của hệ điều hành. Hệ điều hành nachos được thiết kế trong đồ án này được viết bằng ngôn ngữ C được triển khai trên máy linux và giả lập bằng MIPS.

Mô hình giao tiếp chung giữa hệ điều hành và chương trình người dùng:

Diagram

Description automatically generated

Dựa vào mô hình chúng ta sẽ hiểu hơn quy trình hoạt động của nachos. Cách tạo, chạy và biên dịch chương trình trên nachos. Trước khi thực thi một chương trình nào đó trên nachos, trước tiên chúng ta phải biên dịch nachos. Ở đây ta sẽ lấy một chương halt.c làm ví dụ. Chương trình này gọi một hàm duy nhất là Halt() để tắt máy ảo nachos. Qúa trình biên dịch halt() được diễn ra theo các giai đoạn sau:

Giai đoạn 1:  chương trình halt.c được biên dịch bởi cross-compiler gcc thành tập tin halt.s là mã hợp ngữ chạy trên kiến trúc MIPS.

Giai đoạn 2:  tập tin halt.c sẽ được liên kết với tập tin start.s (nằm trong thư mục test) được thay thế cho thư viên C ở mô trình trên để tạo thành halt.coff là định dạng thực thi trên hệ điều hành Linux cho kiến trúc MIPS.

Giai đoạn 3: Chuyển đổi halt.coff thành tập tin halt.noff thông qua file coff2noff.c (trong thư mục bin) để có thể thực thi trên hệ điều hành Nachos cho kiến trúc MIPS.

A screenshot of a computer

Description automatically generated with medium confidence

Sau khi kết thúc quá trình biên dịch trên ta nhận được file halt.noff. Để chạy được chương trình này trên nachos ta dùng lệnh **% .userprog/nachos -rs 1023 -x ./test/halt** . Qúa trình chạy chương trình được thi như sau:

Chương trình start.s luôn được chạy trước halt. Khi vào start.s sẽ chạy \_\_start và được điều hướng nhảy đến hàm main của chương trình bằng lệnh jal main.

Text

Description automatically generated with low confidence

Vì hàm halt() được gọi trong main nên nó sẽ được nhảy đến và gọi đến các syscall có nhãn Halt.

A picture containing diagram

Description automatically generated

Ý nghĩa của dòng **addiu $2, $0, SC\_Halt** là cho mã của system call Halt là SC\_Halt (là 1 số nguyên) vào thanh ghi thứ 2.

Và sau gọi lệnh **syscall** để gọi xử lý system call . Khi nạp từng lệnh , máy ảo giải mã lệnh này , nhận thấy opCode của lệnh là OP\_SYSCALL nó sẽ gọi hàm Raise Exception . Trong hàm này , đơn giản là chỉ chuyển ngắt qua system mode , thực hiện xử lý system call đó , sau đó chuyển lại qua user mode để chạy tiếp chương trình. Hiện tại , trong hàm ExceptionHandle chỉ có xử lý mỗi trường hợp cho system call Halt – tắt máy mà thôi , còn các system call khác thì chưa có.

A picture containing diagram

Description automatically generated

Để viết các system call khác cho hệ điều hành nachos chúng ta viết trong file exception.cc trong hàm ExceptionHandler.

Trong hệ điều hành nachos có 2 chế độ là User Mode và Kernel Mode:

User Mode: vùng nhớ của những chương trình ứng dụng chạy trên NachOS/MIPS

Kernel Mode: vùng nhớ của hệ điều hành NachOS

Do đó, để truyền được các tham số từ chương trình ứng dụng người dùng vào hệ điều hành nachos chúng ta phải thông qua hàm User2System (và System2User trong trường hợp ngược lại) để xử lý.

1. **Cài đặt System call và exception**

Vào thư mục machine vào file machine.h có danh sách các exception được liệt kê. Vào file exception.cc viết lại các case này theo các ExceptionType mà ta bắt được. Mỗi exception ta thêm lệnh interrupt->Halt() để tắt hệ điều hành.

* 1. **Hàm IncreasePC()**

Làm tăng Programming Counter để nạp lệnh tiếp theo để thực hiện. Hàm này thực hiện lưu giá trị của PC hiện tại cho PC trước, đồng thời nạp giá trị kế cho PC hiện tại, và nạp giá trị kế tiếp nữa cho PC kế.

**Graphical user interface, text, application

Description automatically generated**

* 1. **System call ReadInt**

System call ReadInt dùng để đọc một số nguyên mà người dùng nhập từ màn hình console và lưu số nguyên đọc được vào một biến kiểu dữ liệu số nguyên (int) và sau đây là cách mà hàm ReadInt sẽ hoạt động

Text

Description automatically generated

Text

Description automatically generated

* 1. **System call PrintInt**

System call PrintInt dùng để in ra một số nguyên đã được truyền vào như một tham số của hàm. Hàm sẽ in ra số nguyên và sau đây là cách mà hàm PrintInt hoạt động.

Text

Description automatically generated

Text

Description automatically generated

* 1. **System call ReadChar**

System call ReadChar dùng để đọc một kí tự mà người dùng nhập từ màn hình console và lưu kí tự đọc được vào một biến kiểu dữ liệu char và sau đây là cách mà hàm ReadChar hoạt động.



* 1. **System call PrintChar**

System call PrintChar dùng in ra một kí tự đã được truyền vào như một tham số của hàm, hàm sẽ có nhiệm vụ in kí tự ra màn hình console và sau đây là cách hàm PrintChar hoạt động.

Text

Description automatically generated

* 1. **System call ReadString**

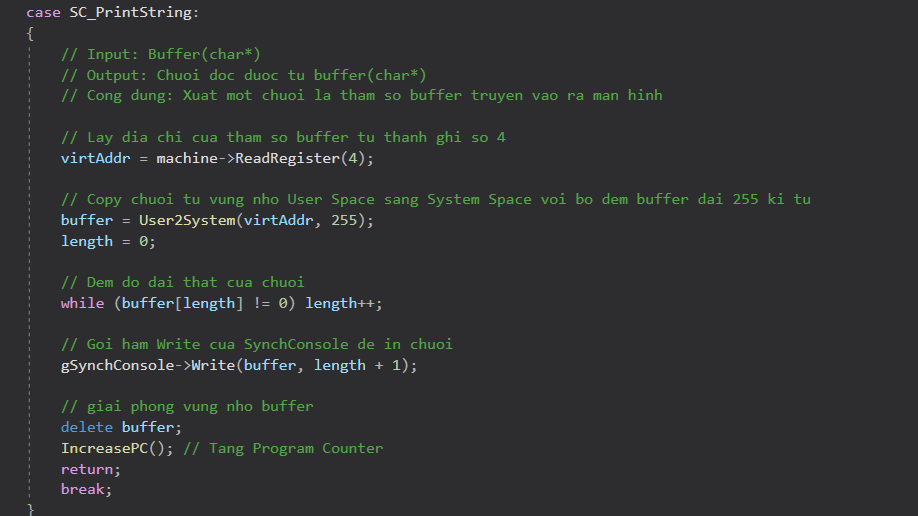
System call ReadString dùng để đọc muột chuỗi kí tự mà người dùng nhập từ màn hình console và lưu kí tự đọc được vào biến kiểu dữ liệu chuỗi (string) và sau đây là cách hoạt động của hàm ReadString.

Text

Description automatically generated

* 1. **System call PrintString**

System call PrintString dùng in ra một chuỗi kí tự đã được truyền vào như một tham số của hàm, hàm sẽ có nhiệm vụ in ra chuỗi kí tự ra màn hình console và sau đây là cách hàm PrintString hoạt động.



1. **Demo minh họa**

Ở file Makefile trong thư mục test, thêm tên các chương trình sau dòng all và thêm đoạn sau vào sau malmult.

****

**…**

**Text

Description automatically generated**

Biên dịch lại nachos, chạy lệnh make

Chạy chương trình bằng lệnh

**./userprog/nachos -rs 1023 -x ./test/<tén chương trình>**

* 1. **Chương trình help**

Chương trình help dùng để in ra các dòng giới thiệu cơ bản về nhóm và mô tả vắn tắt về chương trình sort (mysort) và ascii. Chương trình đơn giản sử dụng system call PrintString(char[]) để in chuỗi ra màn hình console.

**Text

Description automatically generated**

* 1. **Chương trình in bảng ascii**

Chương trình ascii dùng để in ra bảng ascii, sử dụng system call PrintInt và PrintChar để in ra lần lượt mã ascii và ký tự tương ứng.

**Text

Description automatically generated**

* 1. **Chương trình sort (mysort)**

Chương trình sort (đổi tên thành mysort do có file trùng tên trong thư mục test), cho phép người dùng nhập vào một mảng n số nguyên () bằng system call ReadInt và chọn cách sắp xếp chuỗi đã nhập (tăng dần hoặc giảm dần) bằng thuật toán bubble sort và in ra màn hình bằng system call PrintInt

**Text

Description automatically generatedText

Description automatically generated**

1. **Tài liệu tham khảo**