**🙞🙞TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN🙜🙜**

**KHOA CÔNG NGHỆ THÔNG TIN**

**🙣🕮🙡**

**BỘ MÔN HỆ ĐIỀU HÀNH**

ĐỒ ÁN 2:

SYSTEM CALL NACHOS

* Giảng viên: Lê Viết Long
* Sinh viên thực hiện: Võ Hoàng Bảo Duy – 19127027

Trần Ngọc Lam – 19127040

Lê Minh Sĩ – 19127064

C:\Users\tdqua_000\Dropbox\SS-Slides\DeCuong-CDIO\Template CDIO v4.2\Templates\Hinh anh\LogoTruong.png

Khoa Công nghệ Thông tin

Đại học Khoa học Tự nhiên TP HCM

Mục lục

[1. Thông tin chung 3](#_Toc76668367)

[a. Thông tin nhóm 3](#_Toc76668368)

[b. Bảng phân công công việc 3](#_Toc76668369)

[2. Cài đặt tổng quan 4](#_Toc76668370)

[a. Thêm lớp SynchConsole và khởi tạo biến toàn cục 4](#_Toc76668371)

[b. Cài đặt hai thủ tục User2System và System2User 4](#_Toc76668372)

[c. Cài đặt hàm IncreaseProgramCounter() 4](#_Toc76668373)

[d. Cài đặt lại Exception 4](#_Toc76668374)

[e. Hướng dẫn cài đặt System call và chương trình 5](#_Toc76668375)

[3. Cài đặt System Call và Exception 7](#_Toc76668376)

[a. Cài đặt System Call: int ReadInt() 7](#_Toc76668377)

[b. Cài đặt System Call: void PrintInt(int number) 7](#_Toc76668378)

[c. Cài đặt System Call: char PrintChar() 7](#_Toc76668379)

[d. Cài đặt System Call: void PrintChar(char character) 7](#_Toc76668380)

[f. Cài đặt System Call: void ReadString (char[] buffer, int length) 7](#_Toc76668381)

[g. Cài đặt System Call: void PrintString(char[] buffer) 7](#_Toc76668382)

[4. Cài đặt chương trình 7](#_Toc76668383)

[a. Chương trình help 7](#_Toc76668384)

[b. Chương trình ascii 7](#_Toc76668385)

[c. Chương trình sort (Bubble Sort) 7](#_Toc76668386)

[5. Tài liệu tham khảo 7](#_Toc76668387)

# Thông tin chung

## Thông tin nhóm

|  |  |  |
| --- | --- | --- |
| **MSSV** | **Họ Tên** | **Email** |
| *19127027* | Võ Hoàng Bảo Duy | [19127027@student.hcmus.edu.vn](mailto:19127027@student.hcmus.edu.vn) |
| *19127040* | Trần Ngọc Lam | [19127040@student.hcmus.edu.vn](mailto:19127040@student.hcmus.edu.vn) |
| *19127064* | Lê Minh Sĩ | [19127064@student.hcmus.edu.vn](mailto:19127064@student.hcmus.edu.vn) |

## Bảng phân công công việc

|  |  |  |  |
| --- | --- | --- | --- |
| **MSSV** | **Họ Tên** | **Phân công công việc** | **Mức độ hoàn thành** |
| 19127027 | Võ Hoàng Bảo Duy | * Viết báo cáo * Cài đặt tổng quan chương trình và tạo github nhóm * Kiểm tra các lỗi (nếu có) của syscall | 100% |
| 19127040 | Trần Ngọc Lam | * Cài đặt syscall **ReadInt** và **PrintInt** * Viết chương trình **help**, **sort**, **ascii** | 100% |
| 19127064 | Lê Minh Sĩ | * Cài đặt syscall **ReadChar** và **PrintChar** * Cài đặt syscall **ReadString** và **PrintString** | 100% |
| Toàn bộ project | | | 100% |

# Cài đặt tổng quan

## Thêm lớp SynchConsole và khởi tạo biến toàn cục

* Copy 2 tập tin **synchcons.h** và **synchcons.cc** vào thư mục ./code/threads
* Ở thư mục ./code tìm file Makefile.common thực hiện:
  + Cuối đoạn USERPROG\_H thêm **../threads/synchcons.h**
  + Cuối đoạn USERPROG\_C thêm **../threads/synchcons.cc**
  + Cuối đoạn USERPROG\_O thêm **synchcons.o**
* Để hỗ trợ việc nhập/ xuất với màn hình console ta cần sử dụng lớp **SynchConsole**
* Ở thư mục ./code/threads/ thực hiện:
  + Trong file **system.h** ở phần USER\_PROGRAM thêm:

#include "synchcons.h"

extern SynchConsole\* gSynchConsole;

* + Trong file **system.cc:** thực hiện khởi tạo, cấp phát, thu hồi bộ nhớ của biến cục bộ **gSynchConsole**

## Cài đặt hai thủ tục User2System và System2User

* Ở thư **mục ./code/machine** vào file **machine.h** khai báo 2 hàm trong class Machine:
  + **char\* User2System (int virtAddr, int limit)**: có chức năng sao chép vùng nhớ của user sang vùng nhớ của system, kết quả trả về kiểu char\*
  + **int System2User (int virtAddr, int len, char\* buffer):**  có chức năng sao chép vùng nhớ cảu system sang vùng nhớ của user, kết quả trả về là số byte đã sao chép được
* Việc cài đặt 2 hàm trên được thực hiện ở **./code/machine/translate.cc**

## Cài đặt hàm IncreaseProgramCounter()

* Để tránh tình trạng hệ điều hành Nachos bị lặp vô tận khi thực hiện system call ta cần tăng program counter để nạp lệnh tiếp theo trước khi kết quả trả về, công việc này sẽ do hàm **IncreaseProgramCounter** thực hiện.
* Ở thư mục **./code/machine** vào file **machine.h** khai báo hàm trong class Machine
* Việc cài đặt hàm **IncreaseProgramCounter** được thực hiện ở **./code/machine/machine.cc**

## Cài đặt lại Exception

* Ở thư mục ./code/machine vào file **machine.h** để tìm tất cả các exception trong này (hầu hết là lỗi do run-time, không thể phục hồi được).
* Ở thư mục ./code/userprog vào file **exception.cc** thực hiện switch/case với các exception:
  + **No exception:** trả quyền về cho hệ điều hành
  + **SyscallException:** thực hiện switch/ case với từng loại type của System call, sau đó tăng program counter **(machine->IncreaseProgramCounter() )** để nạp lệnh tiếp theo.
  + **Các exception còn lại:** thực hiện việc thông báo người dùng và dừng hệ điều hành ( **interrupt -> Halt()** )
* Các system call tiếp theo được cài đặt và thêm các case vào switch/case của **SyscallException.**

## Hướng dẫn cài đặt System call và chương trình

* Nội dung của các thanh ghi:
  + **R2**: lưu mã system call đồng thời lưu kết quả trả về của system call (nếu có).
  + **R4, R5, R6, R7:** lưu tham số thứ nhất, hai, ba, tư.
* Để tiện cho việc hướng dẫn cài đặt system call và chương trình, ở đây sẽ hướng dẫn cài đặt syscall Sub và chương trình Sub.
* Ở thư mục ./code/userprog vào file syscall.h thêm :

#define SC\_Sub 43

int Sub(int a, int b);

* Ở thư mục ./code/userprog vào file exception.cc thêm case SC\_Sub vào SyscallException:

int op1,op2,result;

op1 = machine->ReadRegister (4);

op2 = machine->ReadRegister (5);

result = op1 - op2;

machine->WriteRegister (2, result);

printf("\nSub is tested");

* Ở thư mục **./code/test** trong các file **start.c** và **start.s** thêm các dòng sau:

.globl Sub

.ent Sub

Sub:

addiu $2,$0,SC\_Sub

syscall

j $31

.end Sub

* Ở phía người dùng, ta sẽ viết chương trình để gọi System Call: thư mục **./code/test:**
  + Thêm file **subTest.c** với đoạn mã gọi hàm Sub:

#include "syscall.h"

int main()

{

int result;

result = Sub(43, 23);

return 0;

}

* + Trong **Makefile:**
    - Thêm **subTest** vào dòng **all**:

all: halt shell matmult sort subTest

* + - Sau đoạn matmult thêm vào đoạn code sau để tạo các file cần thiết để chạy chương trình:

subTest.o: subTest.c

$(CC) $(CFLAGS) -c subTest.c

subTest: subTest.o start.o

$(LD) $(LDFLAGS) start.o subTest.o -o subTest.coff

../bin/coff2noff subTest.coff subTest

* Để chạy test system call và chương trình đã cài đặt, ta biên dịch lại Nachos và chạy chương trình **subTest**. Sau khi chạy chương trình ta sẽ thấy màn hình hiện dòng **“Sub is tested”** – như thế là việc cài đặt system call đã thành công và chương trình cũng đã được chạy.
* Các system call tiếp theo và chương trình theo yêu cầu cũng được thực hiện như các bước trên.

# Cài đặt System Call và Exception

## Cài đặt System Call: int ReadInt()

## Cài đặt System Call: void PrintInt(int number)

## Cài đặt System Call: char PrintChar()

## Cài đặt System Call: void PrintChar(char character)

## Cài đặt System Call: void ReadString (char[] buffer, int length)

## Cài đặt System Call: void PrintString(char[] buffer)

# Cài đặt chương trình

## Chương trình help

## Chương trình ascii

## Chương trình sort (Bubble Sort)

# Tài liệu tham khảo