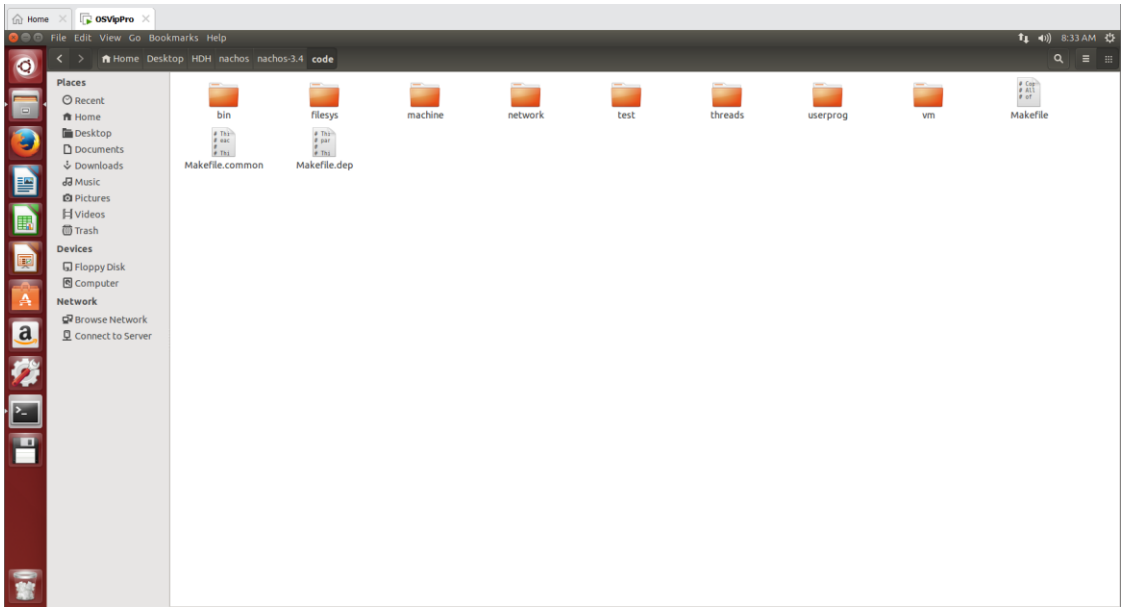
19127057 – Trần Vĩnh Phát
19127108 – Ngô Phú Chiến
19127120 – Ngô Nhật Du

Phần 1. Hiểu mã chương trình NachOS

Nachos là một phần mềm mã nguồn mở (open-source) giả lập một máy tính ảo và một số thành phần cơ bản của hệ điều hành chạy trên máy tính ảo này nhằm giúp cho việc tìm hiểu và xây dựng các thành phần phức tạp hơn của hệ điều hành

- Máy ảo được giả lập có kiến trúc MIPS với hầu hết các thành phần và chức năng của một máy thật như: thanh ghi, bộ nhớ, bộ xử lý, bộ lệnh, chu kỳ thực thi lệnh, cơ chế ngắt, chu kỳ đồng hồ, …
- Hệ điều hành Nachos chạy trên máy ảo Nachos hiện là một hệ điều hành đơn chương
- Được mô phỏng thông qua ngôn ngữ C/C++

Thư mục code của Nachos, chứa hầu hết các đoạn mã nguồn mô phỏng Nachos:



Tất cả những thư mục trên được liên kết với nhau tạo thành một thể thống nhất → Hệ thống Nachos

Phần 2. Hiểu thiết kế

3 thành phần chính của NachOS:

- **Machine:** MIPS (thanh ghi, bộ nhớ, cpu) …/code/machine: chứa các phương thức như Run, ReadRegister, WriteRegister, v.v. Thực hiện xử lý các gián đoạn, hẹn giờ và thống kê.
- **Threads:** …code/threads: chứa các phương thức như Yield, Fork, v.v.
- **UserProgams:** chạy các user program trong address space riêng của họ. Nachos có thể chạy bất kỳ tệp nhị phân MIPS nào nếu nó tự hạn chế chỉ thực hiện các system call của Nachos. Để chuyển sang định dạng đặc biệt của Nachos, dùng CT "coff2noff".

Kernel mode: vùng nhớ của hệ điều hành NachOS
User mode: vùng nhớ của những chương trình ứng dụng chạy trên NachOS/MIPS

Kernel space được dành riêng cho việc chạy nhân hệ điều hành đặc quyền, phần mở rộng nhân và hầu hết các trình điều khiển thiết bị. Ngược lại, **User space** là vùng bộ nhớ nơi phần mềm ứng dụng và một số trình điều khiển thực thi.

• …/code/userprog/exception.cc cài đặt:
char* User2System(int user_buffer, int length): copy vùng nhớ từ user kernel sang system kernel

- Input: địa chỉ vùng nhớ ở user kernel và chiều dài vùng nhớ
- Output: địa chỉ vùng nhớ sau khi copy từ user kernel ở system kernel

int System2User(int user_buffer, int len, char* kernel_buffer): copy vùng nhớ từ system kernel sang user kernel

- Input: địa chỉ vùng nhớ ở user kernel, chiều dài vùng nhớ, địa chỉ vùng nhớ ở system kernel
- Output: chiều dài vùng nhớ thực sự được copy sang user kernel

Các thanh ghi:
• **R2:** Lưu mã và kết quả trả về của mỗi syscall (nếu có)
• **R4, R5, R6, R7**: Lần lượt lưu tham số từ 1 đến 4

Phần 3. Exceptions và system calls

a. .../code/machine/machine.h: ta nhận được các exception sau:

- **NoException**: không có bất kì Exception nào, trả quyền điều khiển về cho operating system

  Mỗi exception sau, ta thêm lệnh "interrupt→Halt()" để dừng (halt) hệ điều hành. Khi các exception này xảy ra thì user program không thể được phục hồi.

- **SyscallException**: được gọi để chuyển sang kernel mode xử lý system call đó và trả về user mode tiếp tục thực thi chương trình. Khi nạp lệnh, máy ảo decode và gọi raise exception (nếu cần).
- **PageFaultException**: Sử dụng cơ chế dịch trang (page-translation) để dịch một địa chỉ tuyến tính sang một địa chỉ vật lý. Operating system báo lỗi và halt hệ thống do tham chiếu đến một address space khi chưa được cấp phát.
- **ReadOnlyException**: Operating system báo lỗi và halt hệ thống do cố gắng thay đổi giá trị cho thuộc tính ưu tiên chỉ đọc ('read-only')
- **BusErrorException:** Operating system báo lỗi và halt hệ thống do virtual address nằm trong kernel space nhưng không tương ứng với bất kỳ physical address hợp lệ nào.
- **AddressErrorException:** Operating system báo lỗi và halt hệ thống do tham chiếu đến một address space không hợp lệ (tham chiếu không được đánh dấu hoặc nằm ngoài phần cuối của address space)
- **OverflowException:** Operating system báo lỗi và halt hệ thống khi arithmetic, ép kiểu hoặc chuyển đổi tác vụ được kiểm tra dẫn đến overflow.
- **IllegalInstrException:** Operating system báo lỗi và halt hệ thống khi hệ thống không ở trạng thái thích hợp cho hoạt động được yêu cầu.
- **NumExceptionTypes**

b.

Viết theo cấu trúc switch case trong …code/userprog/exception.cc

c. .../code/machine/machine.h: tăng program counter để thực hiện lệnh tiếp theo, copy đoạn mã này vào vị trí thích hợp trong phần xử lí các system call.

- **Bước 1:** lưu giá trị thanh ghi hiện tại vào thanh ghi trước đó
- **Bước 2:** đưa thanh ghi hiện tại lên vị trí đoạn lệnh kế tiếp
- **Bước 3:**thanh ghi kế tiếp lên 4 bytes để tới được đoạn lệnh tiếp theo.

d. int ReadInt().

Tham số đầu vào: Không có
Dữ liệu trả về: Trả về số nguyên đọc được từ màn hình console
Chức năng: Đọc số nguyên từ màn hình console

Mô tả:

- **Bước 1:** cấp phát bộ nhớ đệm và có 1 biến length để xác định số byte thực sự đọc được trong bộ nhớ đệm
- **Bước 2:** kiểm tra chiều dài và kí tự đầu tiên để xác định số nguyên đó có hợp lệ hay không.
  Số nguyên đó không hợp lệ nếu: - chiều dài bé hơn 1 hoặc kí tự đầu khác dấu "-"
  - kí tự đầu bé hơn 0 hoặc lớn hơn 9:
  → trả về 0 vào thanh ghi R2, in thông báo "The integer number is not valid"
  thu hồi bộ nhớ cấp phát và tăng program counter
- **Bước 3:** tạo 1 biến để kiểm tra là số dương hay số âm, 1 biến number với giá trị khởi tạo là 0
- **Bước 4:** tiến hành lặp qua từng chữ số
  Kí tự không hợp lệ nếu: bé hơn 0 hoặc lớn hơn 9
  → trả về 0 vào thanh ghi R2, in thông báo "The integer number is not valid"
  thu hồi bộ nhớ cấp phát và tăng program counter
  update biến number
- **Bước 5:** Kiểm tra xem số dương hay số âm, nếu là số âm:
  → biến number sẽ nhân cho -1, và trả kết quả về thanh ghi R2
- **Bước 6:** thu hồi bộ nhớ cấp phát và tăng program counter.

e. void PrintInt(int number)

Tham số đầu vào: Một số interger (điều kiện sẽ được kiểm tra và xử lí trong hàm)
Dữ liệu trả về: không có output
Chức năng: In số nguyên ra màn hình console

Mô tả:

- **Bước 1:** cấp phát bộ nhớ đệm
- **Bước 2:** khởi tạo 1 biến length để xác định chiều dài số nhập
- **Bước 3:** khởi tạo 1 biến num để đọc số nhập vào từ thanh ghi R4 và khởi tạo 1 biến đánh dấu là số âm is_negative
  Nếu biến num = 0
  → bộ nhớ đệm = "0", chiều dài bằng 1
  Num là số âm nếu: num < 0
  → is_negative là đúng (true), đổi num thành số dương
- **Bước 4:** khởi tạo 1 vòng for ghi các chữ số của bộ nhớ đệm theo chiều ngược lại
  vd: số nhập vào là 12345 thì trong buffer hiện tại sẽ là "54321"
- **Bước 5:** khởi tạo 1 vòng for để đảo chiều cho đúng với số nhập vào.
- **Bước 6:** kiểm tra trạng thái của is_negative
  Num là số âm khi: is_negative là true
  → thêm kí tự "-" ở đầu của num
- **Bước 7:** in buffer ra màn hình console
- **Bước 8:** thu hồi bộ nhớ cấp phát và tăng program counter.

f. char ReadChar()

Tham số đầu vào: Không có
Dữ liệu trả về: 1 kí tự char duy nhất
Chức năng: Đọc 1 kí tự char từ màn hình console
Mô tả:

- **Bước 1:** cấp phát bộ nhớ đệm và có 1 biến length để xác định số byte thực sự đọc được trong bộ nhớ đệm
- **Bước 2:** kiểm tra chiều dài và kí tự nhập vào.
  Không hợp lệ nếu: length khác 1
  → trả về 0 vào thanh ghi R2, in thông báo "The input is not valid"
  thu hồi bộ nhớ cấp phát và tăng program counter
  Hợp lệ nếu: length là 1
  → trả về kết quả vào thanh ghi R2
- **Bước 3:** thu hồi bộ nhớ cấp phát và tăng program counter

g. void PrintChar(char character)

Tham số đầu vào: 1 kí tự char duy nhất
Dữ liệu trả về: không có
Chức năng: Xuất một kí tự là tham số arg ra màn hình
Mô tả:

- **Bước 1:** khởi tạo 1 biến c để đọc giá trị từ thanh ghi R4.
- **Bước 2:** dùng gSynchConsole để xuất kí tự đọc được ra màn hình.
- **Bước 3:** tăng program counter và return

h. void ReadString (char[] buffer, int length)

Tham số đầu vào: 1 buffer char[], 1 độ dài chuỗi theo kiểu int
Dữ liệu trả về: Không có
Chức năng: đọc vào 1 chuỗi với tham số là buffer và độ dài tối đa
Mô tả:

- **Bước 1:** cấp phát bộ nhớ đệm cho người dùng (user_buffer) đọc giá trị ở thanh ghi R4 (địa chỉ ở user kernel)
- **Bước 2:** khởi tạo biến length để đọc chiều dài chuỗi từ thanh ghi R5.
- **Bước 3:** dùng hàm User2System để sao chép từ user kernel sang system kernel và chứa tại bộ nhớ đệm cho hệ thống (kernel_buffer)
- **Bước 4:** dùng gSynchConsole để đọc vào system kernel
- **Bước 5:** dùng hàm System2User để ghi ngược từ system ra user kernel.
- **Bước 6:** thu hồi bộ nhớ đệm cho hệ thống, tăng program counter và return.

i. void PrintString (char[] buffer)

Tham số đầu vào: 1 buffer char[]
Dữ liệu trả về: chuỗi đọc được từ buffer char[]
Chức năng: xuất một chuỗi là tham số buffer truyền vào ra màn hình
Mô tả:

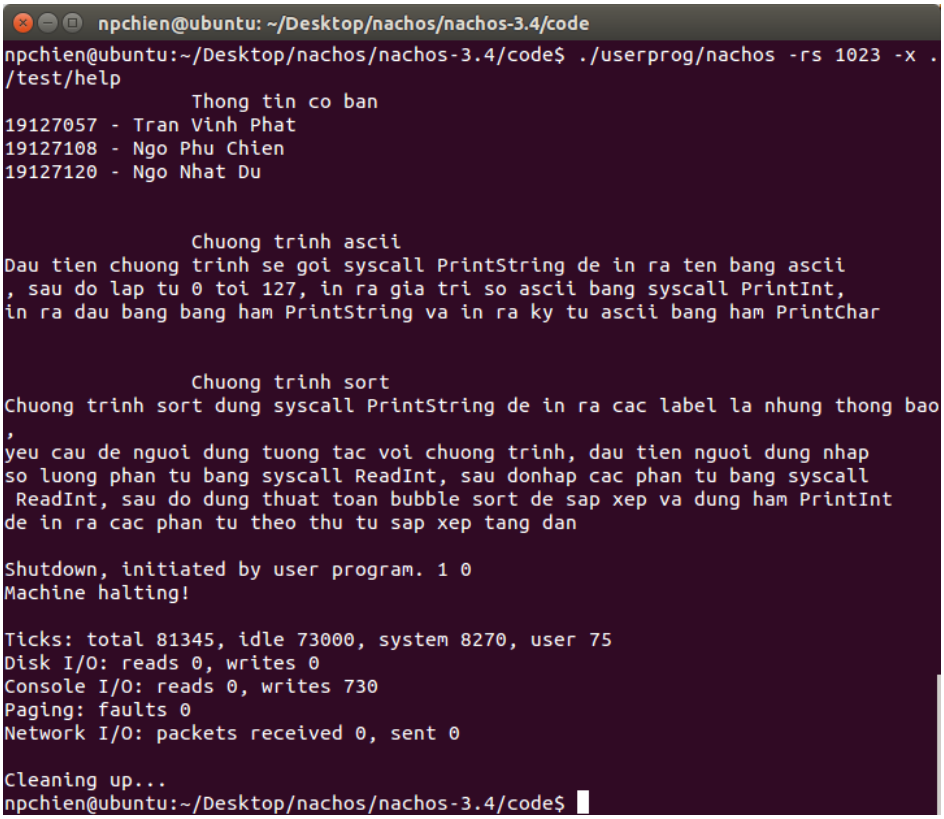
- **Bước 1:** cấp phát bộ nhớ đệm cho người dùng (user_buffer) đọc giá trị ở thanh ghi R4 (địa chỉ ở user kernel)

- **Bước 2:** dùng hàm User2System để sao chép từ user kernel sang system kernel và chứa tại bộ nhớ đệm cho hệ thống (kernel_buffer)
- **Bước 3:** khởi tạo 1 biến length = 0 để tính chiều dài thực sự của chuỗi
- **Bước 4:** update giá trị length
- **Bước 5:** dùng gSynchConsole xuất chuỗi ra màn hình console với chiều dài length + 1
- **Bước 6:** thu hồi bộ nhớ đệm cho hệ thống, tăng program counter và return.

j. Chương trình help

Chức năng: dùng để in ra các dòng giới thiệu cơ bản về nhóm và mô tả vắn tắt về chương trình sort và ascii

Mô tả: gọi system call PrintString(char[]) để in ra những dòng giới thiệu cơ bản về nhóm và mô tả vắn tắt về chương trình sort và ascii



k. Chương trình ascii

Chức năng: in ra bảng mã ascii

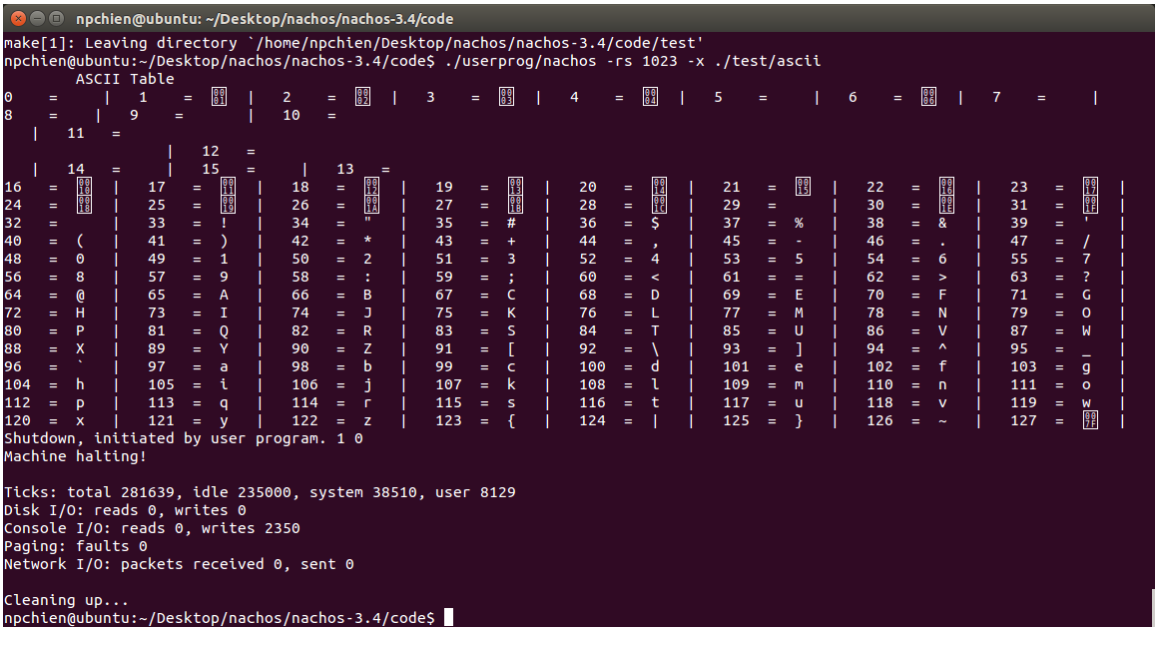
Mô tả:

- **Bước 1:** gọi syscall PrintString để in ra tên bảng ascii
- **Bước 2:** lặp từ 0 đến 127

  gọi PrintInt để in kí tự số

  gọi PrintChar để in kí tự chữ

  gọi PrintString để in kí tự dấu



I. Chương trình sort

Chức năng: sắp xếp mảng được nhập bởi người dùng

Mô tả: chương trình sort dùng syscall PrintString để in ra các label là những thông báo, yêu cầu để người dùng tương tác với chương trình

- **Bước 1:** khởi tạo biến n, gọi system call ReadInt để người dùng nhập số n các phần tử
- **Bước 2:** kiểm tra tính hợp lệ của biến n
  Không hợp lệ nếu: n nằm ngoài khoảng [1; 100]
  → in thông báo "Mảng không hợp lệ" và dừng chương trình
- **Bước 3:** khởi tạo mảng arr, gọi ReadInt() đọc các các kí tự được nhập vào trong arr
- **Bước 4:** dùng thuật toán bubble sort để sắp xếp tăng dần các kí tự trong arr
- **Bước 5:** gọi PrintInt để in ra các phần tử đã sắp xếp theo thứ tự tăng dần