ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN

NachOS - Exceptions và các system calls

Lớp: Hệ điều hành – 20CLC09

Giảng viên hướng dẫn: Gv. Lê Giang Thanh

Gv. Nguyễn Thanh Quân

Gv. Lê Hà Minh

Sinh viên thực hiện:

Võ Ngọc Khánh Vy - 20127099

Nguyễn Thị Thanh Hằng - 20127154

Nguyễn Thị Ngọc Hải – 20127490

TP. Hồ Chí Minh, tháng 4 năm 2022

TABLE OF CONTENTS

I. Mã chương trìnhNachOS:

NachOS là một phần mềm mã nguồn mở giả lập một máy tính ảo cùng một số thành phần cơ bản cuả 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 tính ảo này được giả lập với kiến trúc MIPS, có hầu hết các thành phần và chức năng của một máy thật (thanh ghi, bộ nhớ, bộ xử lý, bộ lệnh, chu kỳ thực thi lệnh, cơ chế ngắt, chu kỳ đồng hồ,..)

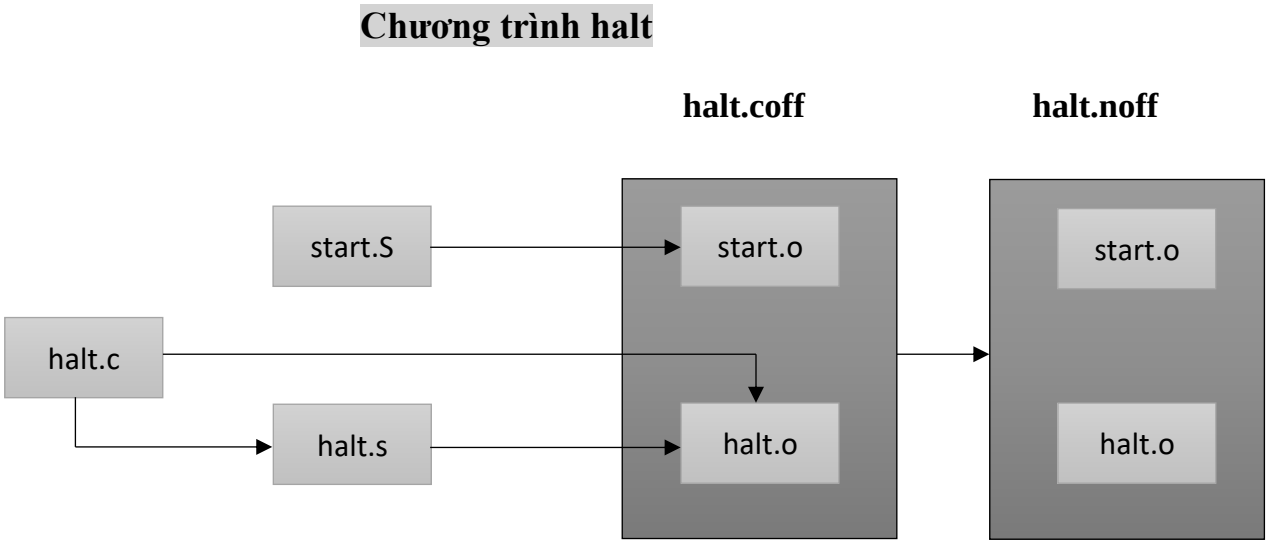
Hệ điều hành NachOS chạy trên máy ảo này là một hệ điều hành đơn chương.

Một số file thành phần và chức năng trong NachOS:

- filesys/: hệ thống quản lý tập tin của NachOS.
- machine/: mô phỏng máy ảo - các thành phần của máy tính khi thực thi chương trình người dùng (bộ nhớ chính – thanh ghi,…)
- test/: chứa chương trình người dùng, biên dịch MIPS, chạy trong NachOS
- threads/: quản lý tiến trình
- userprog/: giao tiếp giữa hệ thống và người dùng (xử lý system call, các exception)
- Các makefile: biên dịch cho C/C++ trong hệ điều hành NachOS
- 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
- bitmap/: các hàm xử lý lớp bitmap
- …

II. Thiết kế:

1. Quá trình thực thi một chương trình trên NachOS



2. **Các bước tạo một system call**

Bước 1 : Khai báo 1 system mới

Bước 2: Thêm dòng lệnh mới

Bước 3: Sửa điều kiện if else thành switch case với các case là các syscall cần gọi

Bước 4: Viết chương trình chạy để người dùng kiểm tra

Bước 5: Thêm đoạn code make file

Bước 6: Biên dịch NachOS

Bước 7: Thực thi chương trình

III. Phân công:

Họ tên	MSSV	Công việc hoàn thành	Mức độ hoàn thành
Võ Ngọc Khánh Vy	20127099	Phần 3: 11 12 13 18 19 20 21 Phần 1, Phần 2, Phần 4	100%
Nguyễn Thị Thanh Hằng	20127154	Phần 3: 3 4 5 6 8 9 17 Phần 1, Phần 2, Phần 4	100%
Nguyễn Thị Ngọc Hải	20127490	Phần 3: 1 2 7 10 14 15 16 Phần 1, phần 2, phần 4	100%

IV. Tiến độ:

Phần	Câu	Ghi chú	Hoàn thành
1		Hiểu mã NachOs	✓
2		Hiểu thiết kế	✓
3	1	Xử lý exceptions	✓
	2	Tăng giá trị PC	✓
	3	ReadNum	✓
	4	PrintNum	✓
	5	ReadChar	✓
	6	PrintChar	✓

	7	Random Num	✓
	8	ReadString	✓
	9	PrintString	✓
	10, 14	Create, Remove	✓
	11	Open, Close	✓
	12	Read	✓
	13	Seek	✓
	15, 16, 17	help, ascii, sort	✓
	18, 19, 20,21	createfile, cat, copy, delete	✓
		Không để user làm sụp hệ điều hành	✓
4		Báo cáo	✓

V. Giới thiệu về exceptions và các system calls:

1. Xử lý các Exception:

Xử lý tất cả các exceptions được liệt kê trong machine/machine.h và hiển thị ra một thông báo lỗi của mỗi exceptions và Halt hệ thống.

2. Tăng Program counter:

Hàm IncreasePC() trong userprog/ksyscall.h có nhiệm vụ tăng program counter để nạp lệnh tiếp theo cần thực hiện. Trước tiên lưu lại giá trị PC hiện tại vào PC trước (PrevPCReg), sau đó tăng giá trị của PC hiện tại lên 4, cuối cùng lưu giá trị của PC kế tiếp (NextPCReg).

InceasePC() được dùng sau khi xử lý xong mỗi exception

3. Chuyển dữ liệu từ User space vào Kernel space:

Hàm User2System(int, int) cài đặt trong file userprog/ksyscall.h với tham số đầu vào là địa chỉ ở user space và độ dài (byte) dữ liệu cần chuyển, trả về buffer dữ liệu sau khi chuyển vào kernel space. Gọi hàm ReadMem() thuộc lớp Machine để đọc lần lượt từng byte trong user space và lưu vào buffer trong kernel space cho đến khi đủ số byte cần chuyển

4. Chuyển dữ liệu từ Kernel space vào User space:

Hàm System2User(it, int, char*) cài đặt trong file userprog/ksyscall.h với tham số đầu vào là địa chỉ ở user space cần chuyển dữ liệu về, độ dài cần chuyển, địa chỉ buffer ở Kernel space cần chuyển đi. Gọi hàm WriteMem() thuộc lớp Machine để ghi lần lượt từng byte dữ liệu lưu trong buffer ở kernel space vào user space.

5. Các System Call:

a. ReadNum

- **Mô tả:**

  void SysReadNum() đọc giá trị số nguyên được nhập từ bàn phím

- **Cách cài đặt:**
  - Đọc lấy dữ liệu vừa nhập từ bàn phím sau đó lưu số lượng kí tự đọc được vào number
  - Xác định dấu (+/-) sau đó kiểm tra số nguyên có phù hợp hay không (xử lý một vài test case), cuối cùng bỏ dấu cho number
  - Kiểm tra số vừa nhập vào có vượt quá giới hạn của INT

b. PrintNum

- **Mô tả:**

  void SysPrintNum() in giá trị số nguyên lưu trong thanh ghi 4 ra màn hình

- **Cách cài đặt:**
  - Kiểm tra dấu của số nguyên đó
  - Chuyển số nguyên thành chuỗi kí tự rồi in ra màn hình

c. ReadChar

- **Mô tả:**

  void SysReadChar() đọc kí tự vừa được nhập từ bàn phím

- **Cách cài đặt:**

  Sử dụng hàm GetChar() trong userprog/synchconsole.cc để đọc kí tự đó vào thanh ghi 2

d. PrintChar

- **Mô tả:**

  void SysPrintChar() In ra màn hình kí tự được lưu trong thanh ghi số 4

- **Cách cài đặt:**

  Sử dụng hàm PutChar(char) trong userprog/synchconsole.cc để in ra kí tự được lưu trong thanh ghi 4

e. RandomNum

- **Mô tả:**

  void SysRandomNum() tạo số nguyên không âm một cách ngẫu nhiên

- **Cách cài đặt:**
  - Sử dụng hàm hỗ trợ RandomInit() khởi tạo trình sinh số ngẫu nhiên
  - Sử dụng hàm RandomNumber() để trả về giá trị nguyên không âm ngẫu nhiên

- Ghi giá trị đó vào thanh ghi 2

f. ReadString:

♦ **Mô tả:**

void SysReadString() đọc chuỗi người dùng vừa nhập từ bàn phím

♦ **Cách cài đặt:**

- Đọc địa chỉ từ thanh ghi số 4, đọc độ dài chuỗi từ thanh ghi số 5
- Sử dụng hàm GetChar() trong userprog/synchconsole.cc để đọc kí tự
- Đọc dữ liệu buffer và chuyển dữ liệu chuỗi buffer vừa đọc được từ Kernel space về User space bằng hàm System2User(int , int, char*)

g. PrintString

♦ **Mô tả:**

void SysPrintString() in chuỗi kí tự ra màn hình

♦ **Cách cài đặt:**

- Đọc địa chỉ từ thanh ghi số 4
- Chuyển dữ liệu trong buffer từ User space sang Kernel space bằng cách sử dụng hàm User2System(int, int)
- Lần lượt in ra màn hình từng kí tự của chuỗi bằng hàm PutChar(char) trong userprog/synchconsole.cc

h. Create:

♦ **Mô tả:**

void SysCreate() tạo file người dùng nhập từ bàn phím.

♦ **Cách cài đặt:**

Lấy địa chỉ file từ thanh ghi R4 -> chuyển giá trị ở R4 từ vùng nhớ user sang system bằng hàm User2System() để lấy tên file. Sau đó kiểm tra file: nếu tên file có độ dài bằng 0 (tức người dùng không nhập) hoặc file không tạo được (file có giá trị bằng NULL hoặc hàm Create() trong fileSystem lỗi) => trả về -1, ngược lại thì trả về 0.

i. Open:

♦ **Mô tả:**

void SysOpen() mở file người dùng nhập từ bàn phím.

♦ **Cách cài đặt:**

Lấy địa chỉ file từ thanh ghi R4 và type từ thanh ghi R5 với type quy ước:

- 0: đọc và ghi.
- 1: chỉ đọc.
- 2: file stdin (đọc từ console).

- 3: file stdout (ghi từ console).

Sau đó lấy tên file bằng hàm User2System().

Gọi hàm FindFreeSlot() trong filesys.h tìm chỗ trống trong mảng openf được khởi tạo với 14 giá trị trong mảng, ứng với vị trí của các file trong openf. Nếu file trống thì phần tử của mảng tại vị trí đó có giá trị bằng NULL.

Dùng switch case để kiểm tra type của file.

Nếu type hợp lệ sẽ ghi giá trị vào thanh ghi R2 tương ứng với mỗi type. Không hợp lệ thanh ghi sẽ có giá trị là -1.

j. Close:

♦ **Mô tả:**

void SysClose() đóng file người dùng nhập từ bàn phím.

♦ **Cách cài đặt:**

Lấy ID file từ thanh ghi R4 -> kiểm tra tính hợp lệ của ID. Nếu hợp lệ và file tồn tại thì xóa đi sau đó gán file có ID đó bằng NULL và trả về kết quả là 0. Ngược lại thì trả về giá trị -1.

k. Read:

♦ **Mô tả:**

void SysRead() đọc nội dung file người dùng nhập từ bàn phím.

♦ **Cách cài đặt:**

Lấy địa chỉ file, số lượng kí tự trong file, id file từ các thanh ghi R4, R5, R6.

- Kiểm tra các điều kiện:
  - ID bé hơn 0 hoặc lớn hơn 14.
  - file trong openf[] có giá trị bằng NULL.
  - type của file có giá trị bằng 3.
- Nếu có 1 trong các điều kiện trên sẽ trả về giá trị -1.
- Nếu điều kiện hợp lệ, gọi hàm GetCurrenPos() trong filesys.h để lấy vị trí con trỏ ban đầu trong file.
- Dùng hàm User2System() để lưu nội dung file vào buffer.
- Xét trường hợp đọc file stdin với type bằng 2, ta gọi phương thức Read của lớp SynchConsoleIn đọc buffer với độ dài numberChar, lấy được số byte cho thanh ghi R2 và chép buffer từ System sang User bằng hàm System2User.
- Đối với file bình thường, lấy vị trí con trỏ hiện tại trong file bằng hàm GetCurrentPos() gọi là Pos, trả về số byte thực sự đọc được cho thanh ghi R2 bằng công thức: newPos – Pos và chép buffer từ phía System sang User bằng hàm System2User().

- Trường hợp file rỗng, trả về -2.

l. Write:

♦ **Mô tả:**

void SysWrite() ghi nội dung file người dùng nhập từ bàn phím.

♦ **Cách cài đặt:**

Tương tự hàm read, lấy địa chỉ, số lượng kí tự, id từ các thanh ghi.

- Sau đó kiểm tra điều kiện:

  - ID bé hơn 0 hoặc lớn hơn 14.

  - file trong openf[] có giá trị bằng NULL.

  - type của file có giá trị bằng 1 hoặc 2.

- Nếu có 1 trong các điều kiện trên sẽ trả về giá trị -1.
- Nếu điều kiện hợp lệ, gọi hàm GetCurrenPos() trong filesys.h để lấy vị trí con trỏ ban đầu trong file.
- Dùng hàm User2System() để lưu nội dung file vào buffer.
- Xét trường hợp type bằng 0, lấy vị trí con trỏ hiện tại trong file bằng hàm GetCurrentPos() gọi là newPos, trả về số byte thực sự ghi được cho thanh ghi R2 bằng công thức: newPos – Pos.
- Trường hợp type bằng 3, gọi hàm Write() của lớp SynchConsoleOut để ghi từng kí tự trong buffer và kết thúc là kí tự xuống dòng '\n', trả về số byte thực sự ghi được cho thanh ghi R2.

m. Seek:

♦ **Mô tả:**

void SysSeek() di chuyển con trỏ tới vị trí trong file người dùng nhập từ bàn phím.

♦ **Cách cài đặt:**

Lấy vị trí của con trỏ trong file và ID file từ thanh ghi R4, R5.

- Kiểm tra ID có hợp lệ hay không.

  - Nếu ID bé hơn 0, lớn hơn 14, bằng 0 hoặc 1 sẽ trả về giá trị -1.

  - Nếu ID hợp lệ, kiểm tra pos có giá trị bằng -1 sẽ gọi hàm Length() để gán lại giá trị cho pos bằng với độ dài của nội dung file (pos lúc này sẽ nằm ở cuối file).

- Kiểm tra pos có giá trị lớn hơn độ dài của nội dung file hoặc bé hơn 0 sẽ trả về giá trị -1.
- Ngược lại, gọi hàm Seek() để di chuyển con trỏ đến vị trí pos và trả về giá trị pos trong thanh ghi R2.

n. Remove:

- **Mô tả:**

  void SysRemove() xóa file người dùng nhập từ bàn phím.

- **Cách cài đặt:**

  Tương tự hàm Create(), lấy địa chỉ file từ thanh ghi R4 -> chuyển giá trị ở R4 từ vùng nhớ user sang system bằng hàm User2System() để lấy tên file. Sau đó kiểm tra file: nếu tên file có độ dài bằng 0 (tức người dùng không nhập) hoặc file không tạo được (file có giá trị bằng NULL hoặc hàm Remove() trong fileSystem lỗi) => trả về -1, ngược lại thì trả về 0

6. Các chương trình:

a. Chương trình help:

Mô tả: Dùng syscall PrintString() để in ra thông tin mô tả thành viên nhóm và lệnh chạy các chương trình khác (Sort, Ascii, Create file,...).

b. Chương trình ascii:

Mô tả: Dùng syscall PrintString(), PrintNum() để thiết kế giao diện bảng mã ascii và syscall PrintChar() dùng để in ra các kí tự từ 32 đến 126 trong bảng mã ASCII vì đây là các kí tự đọc được.

c. Chương trình sort:

Mô tả: Các hàm và công dụng của hàm trong file sort:

- Input(): hàm nhập giá trị các phần tử của mảng từ người dùng
  - Dùng syscall ReadNum() để nhận các giá trị người dùng nhập từ bàn phím lưu vào mảng.
- Print(): hàm in giá trị mảng vừa nhập
  - Dùng syscall PrintNum() để in các phần tử của mảng.
- ascBubbleSort(): hàm sắp xếp các phần tử của mảng bằng thuật toán bubble sort theo thứ tự tăng dần.
- descBubbleSort(): ngược lại với hàm ascBubbleSort(), sắp xếp mảng theo thứ tự giảm dần.
- Hàm main():
  - Cho người dùng nhập số lượng phần tử của mảng bằng syscall ReadNum().
  - Gọi hàm Input() để nhập giá trị của phần tử.
  - Chọn cách sắp xếp mảng theo thứ tự tăng dần hoặc giảm dần.
  - Nếu lựa chọn là 0, gọi hàm ascBubbleSort() để sắp xếp mảng theo thứ tự tăng dần. Ngược lại, lựa chọn là 1 thì gọi hàm descBubbleSort() để sắp xếp theo thứ tự giảm dần.

d. Chương trình createfile:

Mô tả: Chương trình sẽ in ra 2 lựa chọn cho người dùng:

- 1 là tạo file mặc định có tên là 3girls.txt.
- 2 là tạo file có tên do người dùng nhập từ bàn phím.
- Sau khi lựa chọn phương pháp tạo, chương trình sẽ gọi hàm Create() để tạo file theo ý người dùng. Trường hợp nhập lựa chọn khác 1 và 2, chương trình sẽ in ra màn hình "Error mode!"

e. Chương trình cat:

Mô tả: Dùng syscall ReadString() để lưu tên file người dùng nhập.

- Gọi hàm Open() để lấy id file. Kiểm tra id nếu bằng -1 sẽ in ra màn hình "Open file fail".
- Ngược lại, gọi hàm Seek() có giá trị là -1 seek đến vị trí cuối cùng trong file để lấy kích thước nội dung trong file.
- Sau khi có giá trị kích thước, dùng vòng lặp gọi hàm Read() để lưu từng từng kí tự của file và dùng syscall PrintChar() in từng kí tự đó ra màn hình.
- Sau khi kết thúc, gọi hàm Close() để đóng file.

f. Chương trình copy:

Mô tả: Đầu tiên người dùng nhập vào tên file nguồn (file cần copy) và file đích (file ghi nội dung copy).

- Gọi hàm Open() để mở file nguồn và lấy id file. Kiểm tra id file nguồn nếu bằng -1 thì báo lỗi.
- Nếu id file nguồn hợp lệ, gọi hàm Open() để mở file đích và lấy id file đích. Tiếp tục kiểm tra id file đích, nếu file không tồn tại, gọi hàm Create() để tạo file mới và gọi hàm Open() để mở file và lưu lại giá trị id file mới tạo.
- Sau khi mở file đích, gọi hàm Seek() để lấy kích thước nội dung trong file nguồn. Seek về vị trí 0 trong cả 2 file.
- Dùng hàm Read() để đọc kí tự từ file nguồn và hàm Write() để ghi kí tự vừa đọc vào file đích trong vòng lặp với i bé hơn kích thước nội dung file nguồn.
- Sau khi hoàn tất copy, Gọi hàm Close() để đóng 2 file.

g. Chương trình delete:

Mô tả: Để người dùng nhập vào tên file cần xóa.

- Gọi hàm xóa file để kiểm tra file có được xóa thành công hay không.
- Nếu xóa file thất bại, in ra màn hình "Can't delete file". Ngược lại, thông báo xóa thành công.

VI. DEMO:

1. **Integer:**
   - Trường hợp nhập đúng:

```
→  build.linux ./nachos -x ../test/num
Enter a number: 1234
You entered: 1234

Machine halting!
```

```
→  build.linux ./nachos -x ../test/num
Enter a number: -1
You entered: -1

Machine halting!
```

```
→  build.linux ./nachos -x ../test/num
Enter a number: 2147483647
You entered: 2147483647

Machine halting!
```

   - Trường hợp nhập sai:

```
→  build.linux ./nachos -x ../test/num
Enter a number: alo 1 2 3 4
You entered: 0

Machine halting!
```

```
→  build.linux ./nachos -x ../test/num
Enter a number: 2147483648
You entered: 0

Machine halting!
```

```
→  build.linux ./nachos -x ../test/num
Enter a number: 0001
You entered: 1

Machine halting!
```

```
→  build.linux ./nachos -x ../test/num
Enter a number: --1
You entered: 0

Machine halting!
```

- Các system call:

2. **Character:**
   - Trường hợp nhập đúng:

```
→  build.linux ./nachos -x ../test/chr
Enter a char: 1
You entered: 1

Machine halting!
```

   - Trường hợp nhập sai:

```
→  build.linux ./nachos -x ../test/chr
Enter a char: 1 2 3
You entered: 1

Machine halting!
```

```
→  build.linux ./nachos -x ../test/chr
Enter a char: thanhhang
You entered: t

Machine halting!
```

   - Các System call:

3. **Random Number:**

```
→  build.linux ./nachos -x ../test/rand_num
Random number generated: 506924231

Machine halting!
```

   - Các System call:

4. **String**
   - Trường hợp nhập đúng:

```
→  build.linux ./nachos -x ../test/str
Enter a string: thanh hang
You entered: thanh hang

Machine halting!
```

- Trường hợp nhập sai:

```
→  build.linux ./nachos -x ../test/str
Enter a string: ^[[A
You entered:
Machine halting!
```

- Các System call:

5. Help:

```
khanhvy@LAPTOP-QOLEVIU0:~/projects/nachos/NachOS-4.0/code/build.linux$ ./nachos -x ../test/help

                ### Member's group ###
    Vo Ngoc Khanh Vy            -       20127099
    Nguyen Thi Thanh Hang       -       20127154
    Nguyen Thi Ngoc Hai         -       20127490

Introduction:
- Help by using:
" ./nachos -x ../test/help "
- Sorting by using:
" ./nachos -x ../test/sort "
- Ascii table by using:
" ./nachos -x ../test/ascii "
- Create file by using:
" ./nachos -x ../test/create"
- Cat file by using:
" ./nachos -x ../test/cat "
- Copy file by using:
" ./nachos -x ../test/copy "
- Delete file by using:
" ./nachos -x ../test/delete "

Machine halting!
```

6. Sort:

Sắp xếp giảm dần (Bubble sort):

```
khanhvy@LAPTOP-QOLEVIU0:~/projects/nachos/NachOS-4.0/code/build.linux$ ./nachos -x ../test/sort
Enter size of array: 3
Enter elements of array:
A[0] = 0
A[1] = -4
A[2] = 120
0. Ascending
1. Descending
Enter choice: 1
Descending array: 120 0 -4
```

Sắp xếp tăng dần (Bubble sort):

```
khanhvy@LAPTOP-QOLEVIU0:~/projects/nachos/NachOS-4.0/code/build.linux$ ./nachos -x ../test/sort
Enter size of array: 3
Enter elements of array:
A[0] = -5
A[1] = 456
A[2] = 12
0. Ascending
1. Descending
Enter choice: 0
Ascending array: -5 12 456
```

7. **Ascii:**

```
khanhvy@LAPTOP-QOLEVIU0:~/projects/nachos/NachOS-4.0/code/build.linux$ ./nachos -x ../test/ascii

                ==== BANG MA ASCII ====
       Dec     Char    |       Dec     Char    |       Dec     Char
-------------------------------------------------------------------------
        32              |       64      @       |       96      `
        33      !       |       65      A       |       97      a
        34      "       |       66      B       |       98      b
        35      #       |       67      C       |       99      c
        36      $       |       68      D       |       100     d
        37      %       |       69      E       |       101     e
        38      &       |       70      F       |       102     f
        39      '       |       71      G       |       103     g
        40      (       |       72      H       |       104     h
        41      )       |       73      I       |       105     i
        42      *       |       74      J       |       106     j
        43      +       |       75      K       |       107     k
        44      ,       |       76      L       |       108     l
        45      -       |       77      M       |       109     m
        46      .       |       78      N       |       110     n
        47      /       |       79      O       |       111     o
        48      0       |       80      P       |       112     p
        49      1       |       81      Q       |       113     q
        50      2       |       82      R       |       114     r
        51      3       |       83      S       |       115     s
        52      4       |       84      T       |       116     t
        53      5       |       85      U       |       117     u
        54      6       |       86      V       |       118     v
        55      7       |       87      W       |       119     w
        56      8       |       88      X       |       120     x
        57      9       |       89      Y       |       121     y
        58      :       |       90      Z       |       122     z
        59      ;       |       91      [       |       123     {
        60      <       |       92      \       |       124     |
        61      =       |       93      ]       |       125     }
        62      >       |       94      ^       |       126     ~
        63      ?       |       95      _       |

Machine halting!
```

VII. Tài liệu tham khảo:

1. Tài liệu trên moodle do giảng viên cung cấp
2. https://www.youtube.com/playlist?list=PLRgTVtca98hUgCN2_2vzsAAXPiTFbvHpO
3. https://github.com/nguyenthanhchungfit/Nachos-Programing-HCMUS