# BÁO CÁO

# ĐỒ ÁN 01

# Exceptions và các System calls đơn giản

## A. Thông tin thành viên

| MSSV | Họ và tên | Mức độ đóng góp |
|---|---|---|
| 19120036 | Nguyễn Đăng Tiến Thành | 100% |
| 19120176 | Lê Công Bình | 100% |
| 19120200 | Nguyễn Tam Dũng | 100% |

## B. Cài đặt Exceptions và System calls

### 1. Xử lý các exception

Cài đặt hết danh sách các exception được liệt kê trong enum *ExeptionType* ở file "*machine/machine.h*". Với mỗi exeption bắt được in ra mô tả thông báo lỗi và gọi SysHalt() để tắt hệ điều hành.

### 2. Tăng Program counter

Hàm IncreasePC() trong *"exception.cc"* 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 dùng cho debug), sau đó tăng giá trị của PC hiện tại lên 4, cuối cùng lưu lại giá trị của PC kế tiếp (NextPCReg dùng cho rẽ nhánh).

IncreasePC() được dùng sau khi xử lý xong mỗi exception.

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

Hàm User2System() cài đặt trong file "*exeption.cc*" 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.

Sử dụng 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 về User space

Hàm System2User() 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.

Sử dụng hàm WriteMem() thuộc lớp Machine để ghi lần lượt từng byte dữ liệu lưu trong buffer ở kernel space vào user space.

## 5. Cài đặt các system call

a. ReadNum

- Mô tả system call SC_ReadNum: int ReadNum()
    - Input: không có.
    - Output: giá trị số nguyên được nhập từ console.
    - Mục đích: Đọc giá trị số nguyên được nhập từ bàn phím.
- Cách cài đặt:
    - Dùng hàm kernel->synchConsoleIn->GetChar() để đọc từng kí tự trên console vào để xử lí
    - Dùng 1 vòng lặp while để loại bỏ hết các kí tự khoảng trắng dư thừa. Vòng lặp này kết thúc khi gặp kí tự khác ' ' và '\n'.
    - Nếu kí tự đầu tiên là dấu '-' thì đánh dấu là số âm (gắn *isSign = true*) và bắt kí tự tiếp theo. Nếu là kí tự ' ' và '\n' thì trả về 0 ngay, còn nếu là kí tự khác chữ số thì đánh dấu đây không phải là số nguyên.

- Dùng thêm 1 vòng lặp để nhận các kí tự tiếp theo và chuyển thành giá trị số (giá trị tuyệt đối), nếu gặp kí tự khoảng trắng ‘ ‘ hoặc ‘\n’ thì kết thúc nhập số. Nếu 1 kí tự không phải chữ số thì sẽ đánh dấu đây không phải là số nguyên (*isInteger = false*). Khi vòng lặp kết thúc, nếu isInteger có giá trị false thì trả về 0 ngay. Làm vậy để tránh việc ngừng bắt kí tự giữa chừng khiến các kí tự còn lại nhảy xuống dòng lệnh tiếp theo, chỉ dừng và trả về kết quả khi người dùng đã nhấn ‘\n’ hoặc ‘ ‘.
- Trước khi trả về kết quả kiểm tra biến isSign, nếu true thì sẽ lấy giá trị âm của kết quả, nếu false thì giữ nguyên.
- Ghi kết quả đọc được vào thanh ghi số 2

b. PrintNum

- Mô tả system call SC_PrintNum: void PrintNum(int)
  - Input: số nguyên cần in.
  - Output: không có.
  - Mục đích: In giá trị số nguyên ra màn hình console.
- Cách cài đặt:
  - Đọc số nguyên cần in từ thanh ghi số 4
  - Đầu tiên, kiểm tra xem có phải là số 0 hay không, nếu có, in ra và thoát chương trình. Mục đính là tránh việc khởi tạo một mảng kí tự 11 phần tử nếu như không cần thiết.
  - Kiểm tra xem có là số âm hay không, nếu có thì sẽ in ra dấu ‘-’ đầu tiên, sau đó chuyển biến thành giá trị tuyệt đối của nó.
  - Tiến hành khởi tạo một mảng char[10], dùng vòng lặp while để chuyển dần từng kí tự trong số vào mảng. Trong mỗi vòng lặp, r sẽ bằng số chia lấy dư cho 10, phần tử thứ i của mảng sẽ lấy giá trị chia dư đó. Sau mỗi lần lặp, biển i sẽ được tăng giá trị lên 1,

còn số cần in thì bị chia nguyên cho 10. Lý do chỉ sử dụng mảng char[10] vì số int dài nhất ($2^{31}$ - 1) chỉ tối đa 10 kí tự thập phân.

- Vòng lặp for tiếp theo sẽ in các kí tự đã ghi lại trong mảng ra console.

c. ReadChar

- Mô tả cài đặt SC_ReadChar: char ReadChar()
  - Input: không có
  - Output: kí tự nhập từ console.
  - Mục đích đọc kí từ nhập từ bàn phím
- Cài đặt:
  - Dùng synchConsoleIn để đọc một kí tự người dùng nhập.
  - Ghi kết quả đọc được vào thanh ghi số 2

d. PrintChar

- Mô tả cài đặt SC_PrintChar: void PrintChar(char c)
  - Input: kí tự c.
  - Output: không có.
  - Mục đích: in kí tự input ra màn hình console.
- Cài đặt:
  - Đọc kí tự cần in từ thanh ghi số 4
  - Dùng SynchConsoleOut để in một kí tự từ biến char c truyền vào ra màn hình console.

e. RandomNum

- Mô tả system call SC_RandomNum:
  - Input: không có
  - Output: một số nguyên không âm ngẫu nhiên
  - Mục đích: sinh ngẫu nhiên số nguyên không âm
- Cài đặt:

- Sử dụng hàm hỗ trợ sẵn RandomInit() trong *"lib/sysdep.h"* để khởi tạo trình sinh số ngẫu nhiên
- Sử dụng hàm RandomNumber() trong *"lib/sysdep.h"* để sinh ngẫu nhiên một số nguyên không âm

f. ReadString

- Mô tả system call SC_ReadString
  - Input: địa chỉ buffer lưu string cần đọc, độ dài của string
  - Output: không có
  - Mục đích: cho phép người dùng nhập một chuỗi từ console và lưu trữ vào mảng kí tự buffer
- Cài đặt:
  - Đọc địa chỉ của tham số *buffer* từ thanh ghi số 4, và đọc độ dài chuỗi nhập vào *length* từ thanh ghi số 5
  - Chuyển dữ liệu của *buffer* từ User space vào Kernel space thông qua hàm User2System()
  - Thực hiện đọc dữ liệu, bằng cách sử dụng vòng lặp và gọi hàm GetChar() của lớp SynchConsoleIn để đọc đủ *length* kí tự
  - Sau khi đọc xong, chuỗi đang nằm ở Kernel space do đó, cần chuyển dữ liệu chuỗi *buffer* vừa đọc được từ Kernel space về User space thông qua hàm System2User()

g. PrintString

- Mô tả system call SC_PrintString
  - Input: địa chỉ chuỗi *buffer* cần in ra console
  - Output: không có
  - Mục đích: dùng để in chuỗi kí tự trong buffer ra màn hình
- Cài đặt:
  - Đọc địa chỉ tham số *buffer* từ thanh ghi số 4

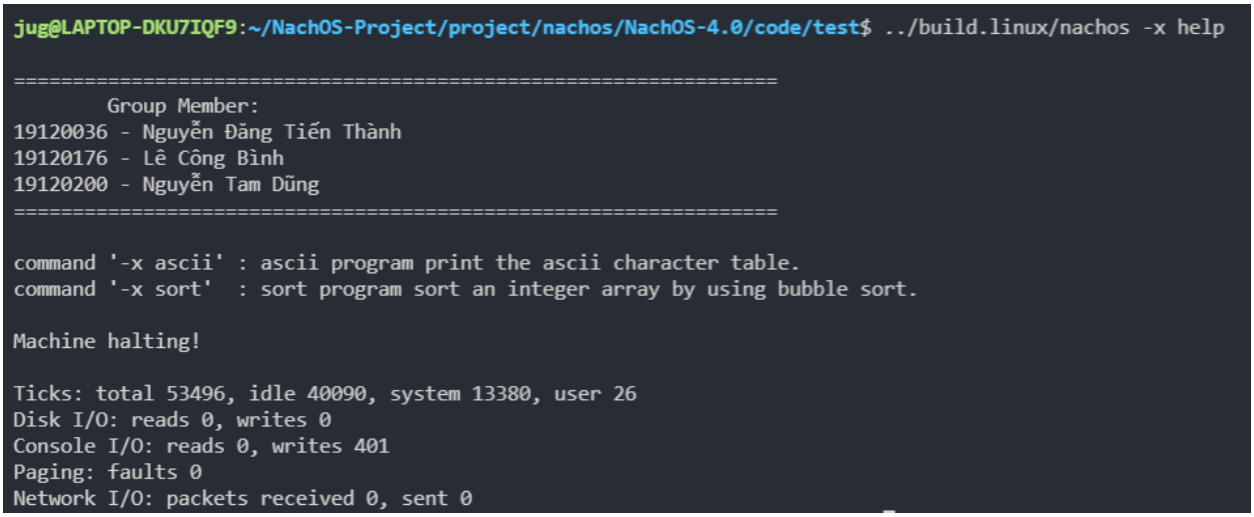
- Chuyển dữ liệu trong buffer từ User space vào Kernel space
- Sau khi đã chuyển dữ liệu vào Kernel space, sử dụng vòng lặp và hàm PutChar() trong lớp SynchConsoleOut để in ra từng kí tự cho đến khi gặp kí tự kết thúc '\0'

## C. Demo minh họa

### 1. Minh họa các system call đã cài đặt


Demo các system call đã cài đặt: ReadString, PrintString, ReadChar, PrintChar, ReadNum, PrintNum và RandomNum

### 2. Chương trình help




Chương trình help sử dụng system call PrintString để in các thông tin về nhóm và mô tả về chương trình ascii và sort.

## 3. Chương trình ascii

```
congbinhle@pop-os:~/Projects/HCMUS/NachOS-Project/project/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x 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!

Ticks: total 109277, idle 78290, system 26460, user 4527
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 783
Paging: faults 0
Network I/O: packets received 0, sent 0
congbinhle@pop-os:~/Projects/HCMUS/NachOS-Project/project/nachos/NachOS-4.0/code/test$
```

Chương trình sử dụng hàm system call PrintChar để in bảng mã ASCII giá trị từ 32 tới 126, vì đây là các kí tự đọc được. Để dễ quan sát, chương trình được viết để in bảng mã thành 3 cột, mỗi cột bao gồm giá trị thập phân cùng với kí tự tương ứng trong bảng mã ASCII.

## 4. Chương trình sort

- **Sắp xếp tăng dần**

```
jug@LAPTOP-DKU7IQF9:~/NachOS-Project/project/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x sort
Enter number of elements n = 8
Enter array: 1 7 9 12 66 1 100 21
Ascending/Descending enter [0/1]: 0
Array after sorting: 1 1 7 9 12 21 66 100
Machine halting!

Ticks: total 2081552178, idle 2081545170, system 4990, user 2018
Disk I/O: reads 0, writes 0
Console I/O: reads 25, writes 119
Paging: faults 0
Network I/O: packets received 0, sent 0
```

- **Sắp xếp giảm dần**

```
jug@LAPTOP-DKU7IQF9:~/NachOS-Project/project/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x sort
Enter number of elements n = 8
Enter array: 12 1 7 9 72 27 43 2
Ascending/Descending enter [0/1]: 1
Array after sorting: 72 43 27 12 9 7 2 1
Machine halting!

Ticks: total 1106754585, idle 1106747210, system 4950, user 2425
Disk I/O: reads 0, writes 0
Console I/O: reads 24, writes 118
Paging: faults 0
Network I/O: packets received 0, sent 0
```

Chương trình sort sử dụng system call ReadNum để đọc mảng số nguyên do người dùng nhập vào, sử dụng system call ReadChar để nhận biết yêu cầu sắp xếp theo thứ tự tăng dần (nhập 0) hay giảm dần (nhập 1). Cuối cùng sử dụng system call PrintNum để in ra mảng sau khi sắp xếp.