

Operating System

NachOS - Final Project Report

Team Members

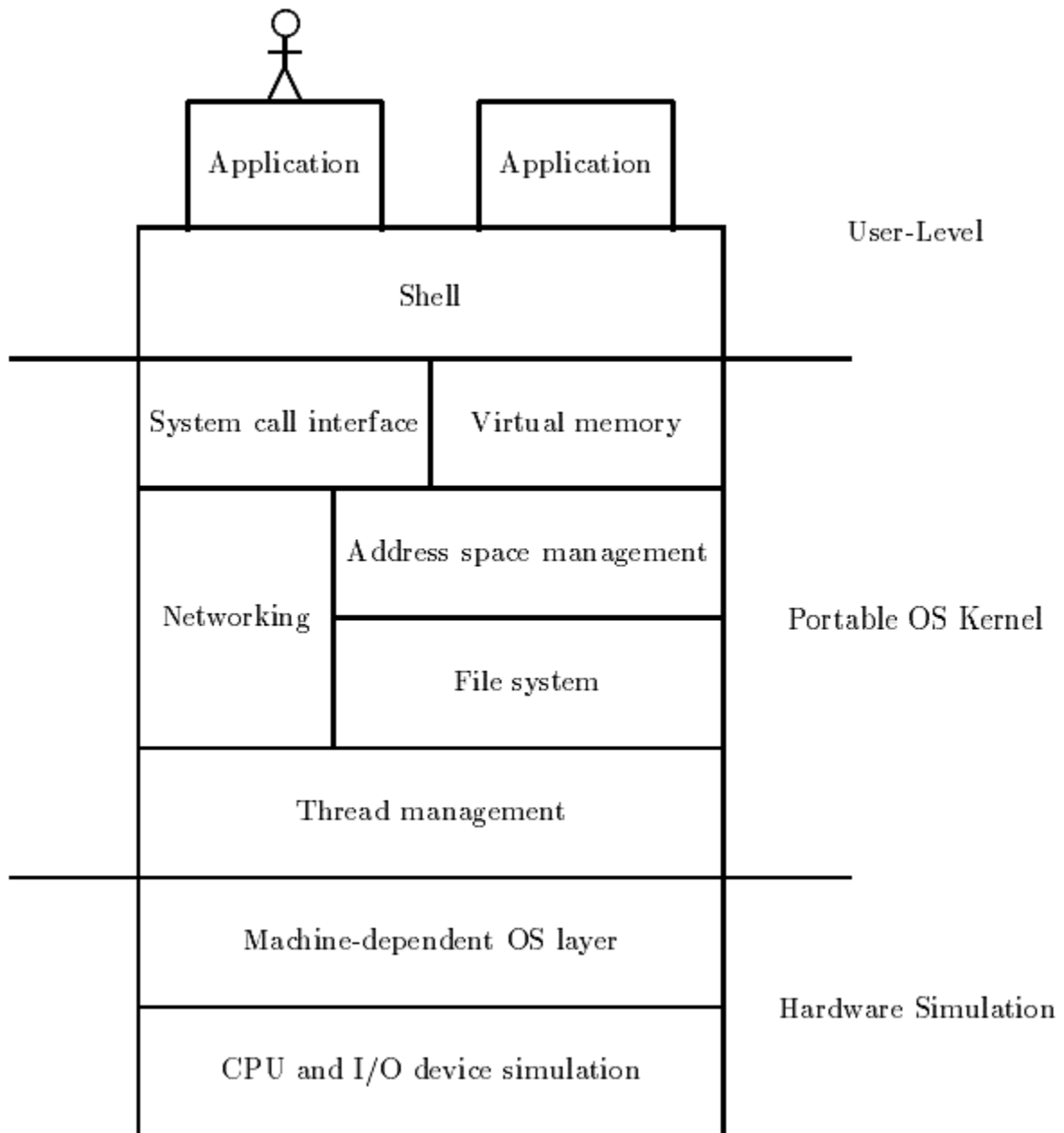
Neel Patel #201501075

Prasann Patel #201501083

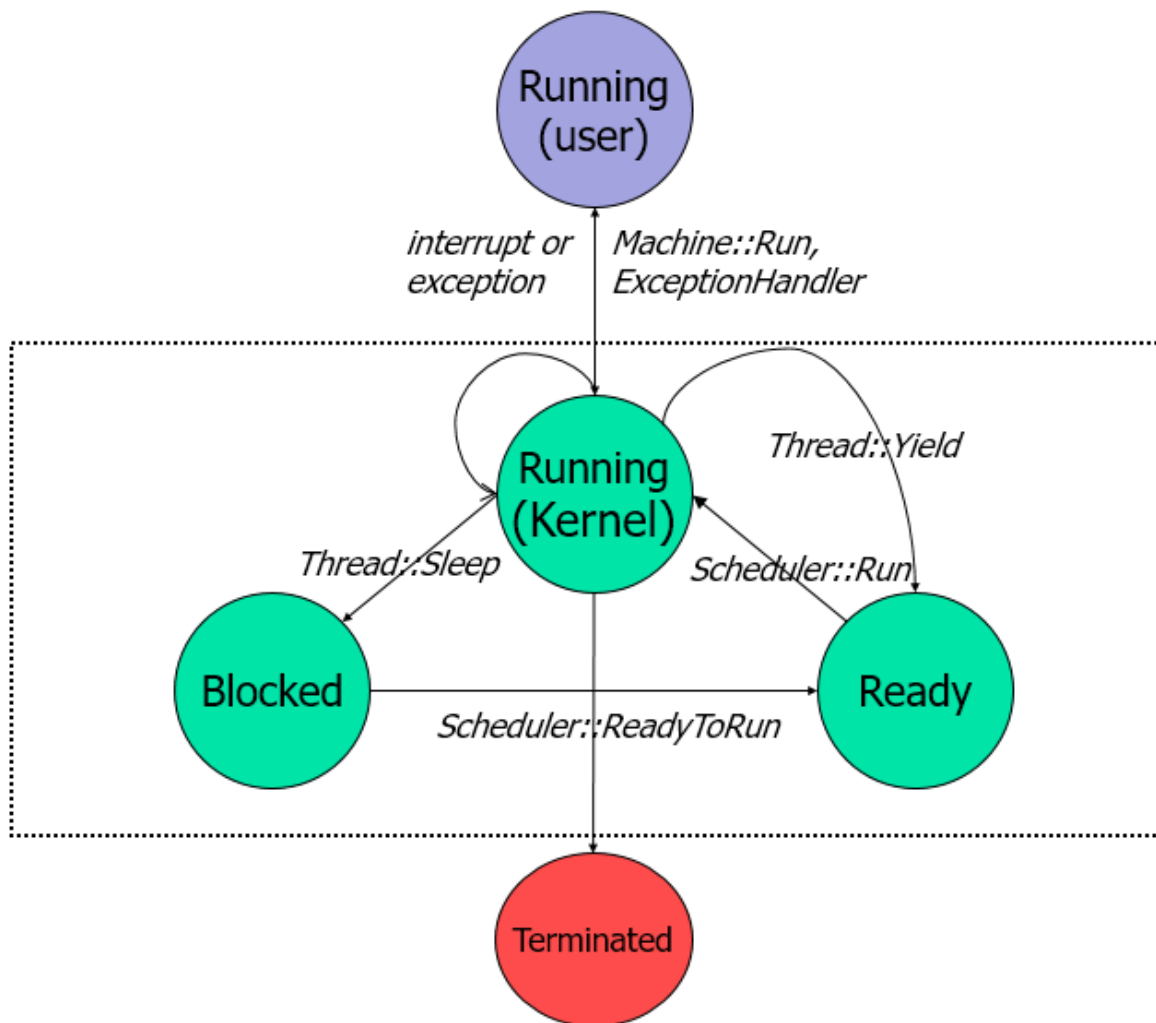
Description

Not Another Completely Heuristic Operating System (NachOS) is a multitasking instructional system that runs on a UNIX process which illustrates and explores all areas of modern operating systems including threads and concurrency, multiprogramming, system calls, software-loaded TLB's, file systems, and distributed systems.

Architecture



Model/Diagram(s)



Technical Specification

NachOS uses C++/JAVA and basic MIPS instructions to run as a user-process on top of host OS like Ubuntu-Linux. NachOS interacts with simulated H/W by calling functions that eventually calls underlying host OS library routines. Also, in NachOS' interrupts happen only as discrete points in real time as time is also simulated.

Scope of the project includes designing and implementing features like :

1. Thread management - Thread Control Block (TCB.contextSwitch)
2. Concurrency and Synchronization
3. Interrupt Handling
4. Multiprogramming - 50%
5. Filling services (i/o Console)

Algorithm/Flow Chart

Pseudo-code for different functions in Kthread are as follows :

```
join(){
    Disable interrupts;
    if (join Queue not be initiated) {
        create a new thread queue (join Queue) with transfer priority flag opened
        join Queue acquires this thread as holder
    }
    If (CurrentThread != self) and (status is not Finished) {
        add current thread to join queue
        sleep current thread
    }
    Re-enable interrupts;
}

finish(){
    disable Interrupts.
    Destroy current thread
    Assign status = finished
    Sleep forever }

sleep(){

    release condition lock
    disable interrupt
    add current thread to wait queue
    make current thread sleep
    restore interrupt
    acquire condition lock
}
```



wake(): wake up a single thread sleeping in this condition variable, if possible.

if wait queue is not empty
 disable interrupt
 remove the first element from wait queue
 put the first element into ready queue
 restore interrupt

yield(): Relinquish the CPU if any other thread is ready to run. If so put the current thread on the ready queue which will be re-scheduled later.

Store status of current thread
If wait queue is not empty
 Disable interrupts
 Send current thread to ready queue
 Invoke runNextThread()
 restore interrupt

run(): Dispatch the CPU to this thread. Save the state of the current thread, switch to the new thread by calling TCB.contextSwitch() & load the state of the new thread. The new thread becomes the current thread.

Note : Ready never adds the idle thread to the running state [Eg. thread 1 (#ideal thread)]

Change state of previously running thread/ps.
disable interrupts
invoke Machine.yield()
invoke tcb.contextSwitch()
restore state

KThread has status – New, Ready, Running, Blocked, Finished

Program and Test Data Set

Included in the source code file :

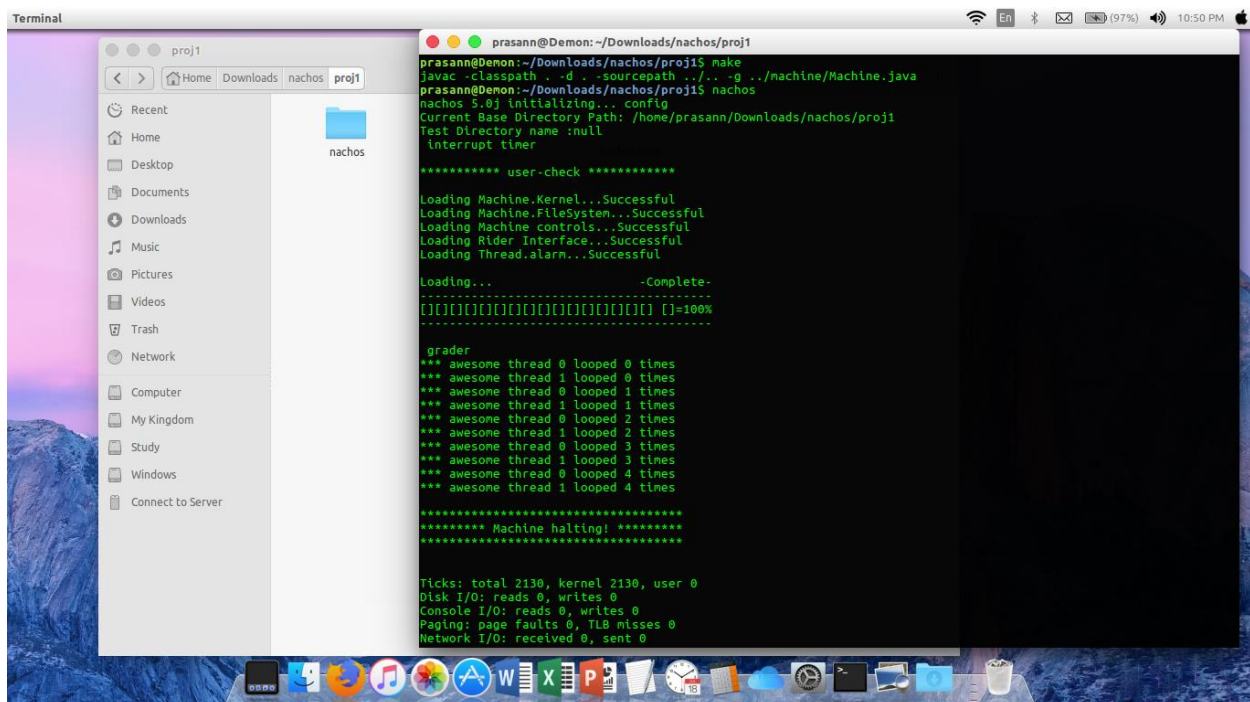
Test input :

1. `cd nachos/proj#`
2. `make`
3. `nachos <switch\optional> <arguments>`

Eg. `nachos -s 10` (random thread placement)
`nachos -d t` (to debug)
`nachos` (simple exe.)
`nachos -np` (About the developers and project)

Results

Proj1: Thread Management



The screenshot shows a macOS desktop environment. On the left, a file browser window titled 'proj1' is open, displaying the contents of the 'nachos' directory. On the right, a terminal window titled 'prasann@Demon: ~/Downloads/nachos/proj1' is running the 'nachos' program. The terminal output shows the program's initialization, including loading the Machine.Kernel, Filesystem, controls, Rider Interface, and Thread.alarm. It then displays a progress bar for 'Loading...' which is 100% complete. Following this, it shows a 'grader' section with a list of 'awesome thread' entries, each with a looped count. The program concludes with 'Machine halting!' and a summary of system statistics: 'Ticks: total 2130, kernel 2130, user 0', 'Disk I/O: reads 0, writes 0', 'Console I/O: reads 0, writes 0', 'Paging: page faults 0, TLB misses 0', and 'Network I/O: received 0, sent 0'.

```
Terminal
proj1
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Network
Computer
My Kingdom
Study
Windows
Connect to Server

nachos

prasann@Demon: ~/Downloads/nachos/proj1
prasann@Demon:~/Downloads/nachos/proj1$ make
javac -classpath . -d . -sourcepath ../.. -g ../machine/Machine.java
prasann@Demon:~/Downloads/nachos/proj1$ nachos
nachos 5.0j initializing... config
Current Base Directory Path: /home/prasann/Downloads/nachos/proj1
Test Directory name :null
Interrupt timer

***** user-check *****

Loading Machine.Kernel...Successful
Loading Machine.FileSystem...Successful
Loading Machine.controls...Successful
Loading Rider Interface...Successful
Loading Thread.alarm...Successful

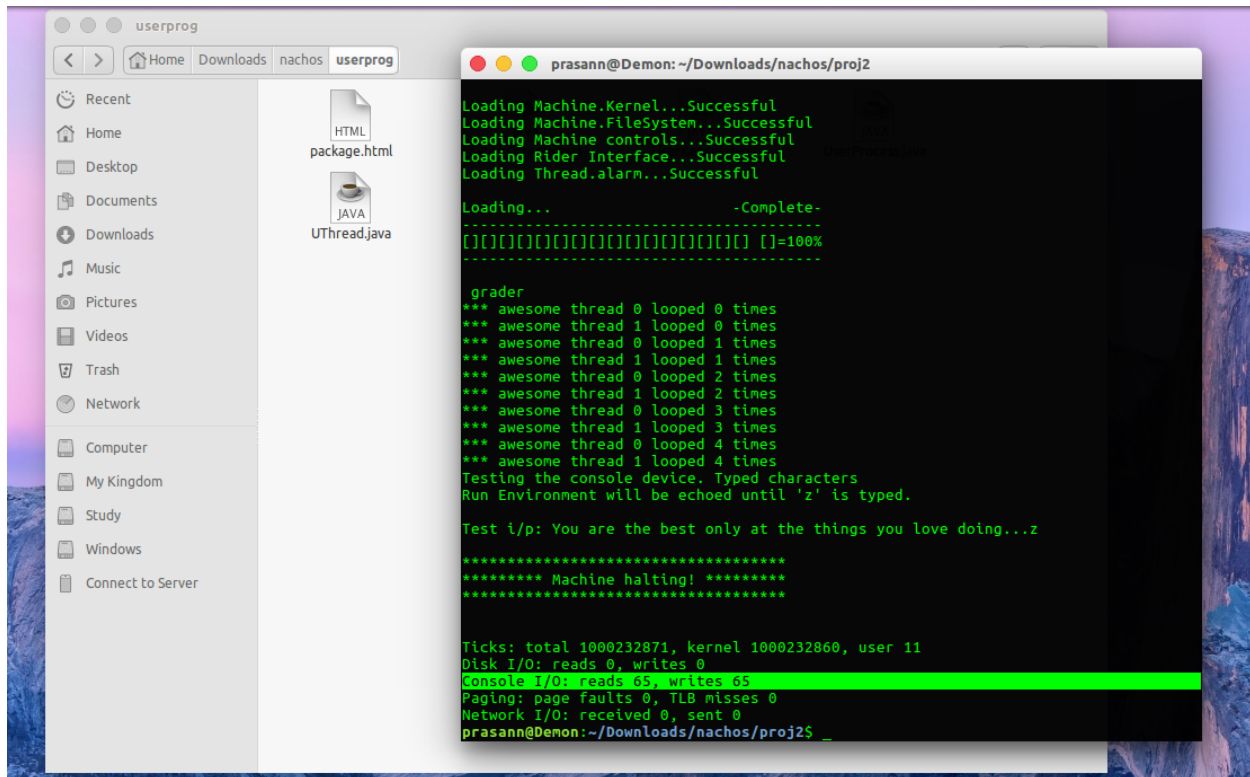
Loading... -Complete-
[ ]=100%

grader
*** awesome thread 0 looped 0 times
*** awesome thread 1 looped 0 times
*** awesome thread 0 looped 1 times
*** awesome thread 1 looped 1 times
*** awesome thread 0 looped 2 times
*** awesome thread 1 looped 2 times
*** awesome thread 0 looped 3 times
*** awesome thread 1 looped 3 times
*** awesome thread 0 looped 4 times
*** awesome thread 1 looped 4 times

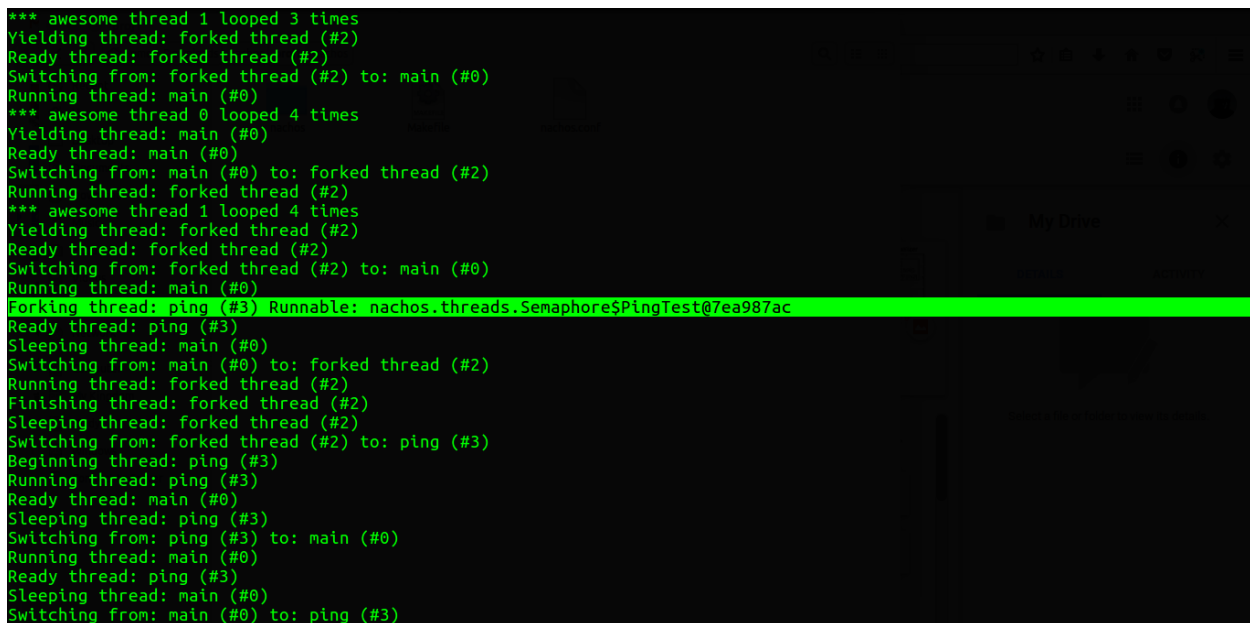
***** Machine halting! *****

Ticks: total 2130, kernel 2130, user 0
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: page faults 0, TLB misses 0
Network I/O: received 0, sent 0
```

Proj2: Multi-programing



Debug mode :





Implementation/Source Code

Attached in Code file.

References

1. *The Nachos Instructional Operating System* by Wayne A.Christopher, Steven J.Procter and Thomas E.Anderson -Computer Science Division -University of California - Berkeley
2. *University of California, San Diego : CSE 120 Nachos Project Guide - Fall 2017*
<http://cseweb.ucsd.edu/classes/fa17/cse120-ab/projects/index.html>
3. CS162 Project Phase 1 : <http://inst.eecs.berkeley.edu/%7Ecs162/fa10/Nachos/phase1.html>

