國立臺灣大學 National Taiwan University

Nachos

EE 5173 Operating System

TA: Hsueh-Yi, Chen. Advisor: Farn, Wang.

2023/9/21

Outline

- Nachos Overview
- Installation
- Project 1
  - Part1: Multi-Programming
  - Part2: System call tracing

The history

- Around 1980s, numerous projects have been developed for teaching OS, most of them were motivated by UNIX.
- Thanks to the technology growth, we can execute an OS kernel as an UNIX process using a simulation of real hardware.
- Advancement in OS, hardware and software had left many OS projects out of date.
  -> This is why Nachos came in.

Nachos overview

- Not Another Completely Heuristic Operating System

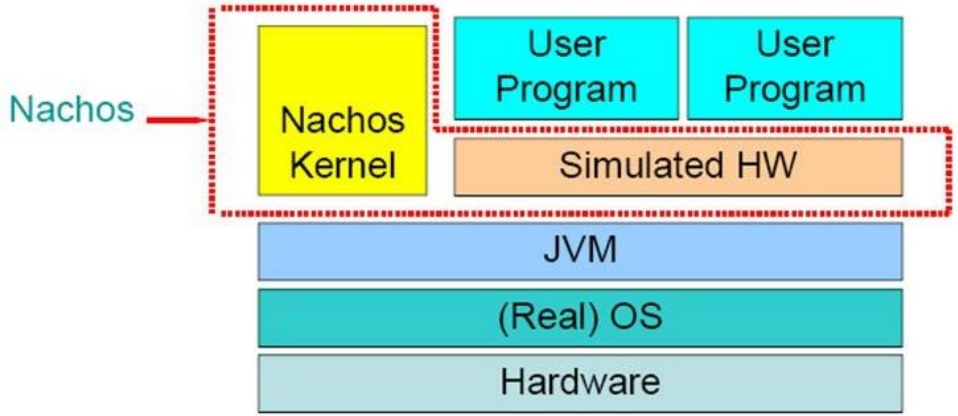
  https://homes.cs.washington.edu/~tom/nachos/

- 1992 (31 years ago), Thomas Anderson & his students. UC Berkeley
- Written in **C++,** but Danial Hattena has rewritten Nachos in JAVA
- Instructional software for undergraduate & graduate student.
- Simulate MIPS R2/3000 machine

Nachos overview

- Only difference between Nachos & real OS is that Nachos runs as a single Unix process, real OS run on bare machines.

✓ Bare machine: hardware which is used to execute the program in the processor without using the operating system

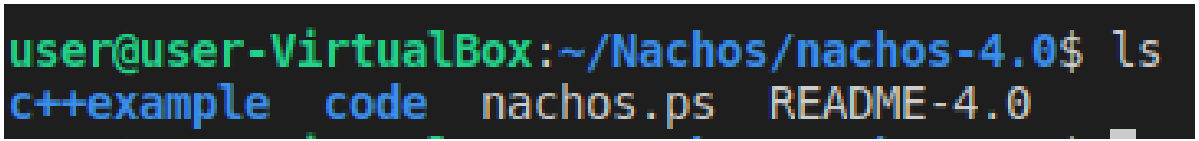


Some Statistics

- 8 homework: Context Switching, semaphores, Address Translation…
- 4 project: thread management, Multiprogramming, virtual memory/file systems, networking
- Use around the world: Duke University, University of Washington, UC Berkely, University of Waterloo, New York University…

https://homes.cs.washington.edu/~tom/cs162sp96/

Directory Structure



- c++example: some c++ introduction and examples
- Code : Nachos source code
- Nachos.ps: nachos introduction (postscript)

cd code

Directory Structure

- **machine/**

  - hardware simulation code

- **lib/**

  - utilities function of nachos, some data structure(bitmap, hash, list…)

- **threads/**

  - nachos is also multi-thread support, there are also entry point routine main in the folder

Directory Structure

- **test/**

  - test user program and written in c

- **userprog/**

  - Nachos operating system code. (e.g., create an address space, load, execute, exception handling)

- **network/**

  - post (deliver incoming network messages to mailbox (buffer))

  - netkernel (routine for the network communication support kernel, kerneltype)

Directory Structure

- **filesys/**

  - 2 version nachos filesystem.

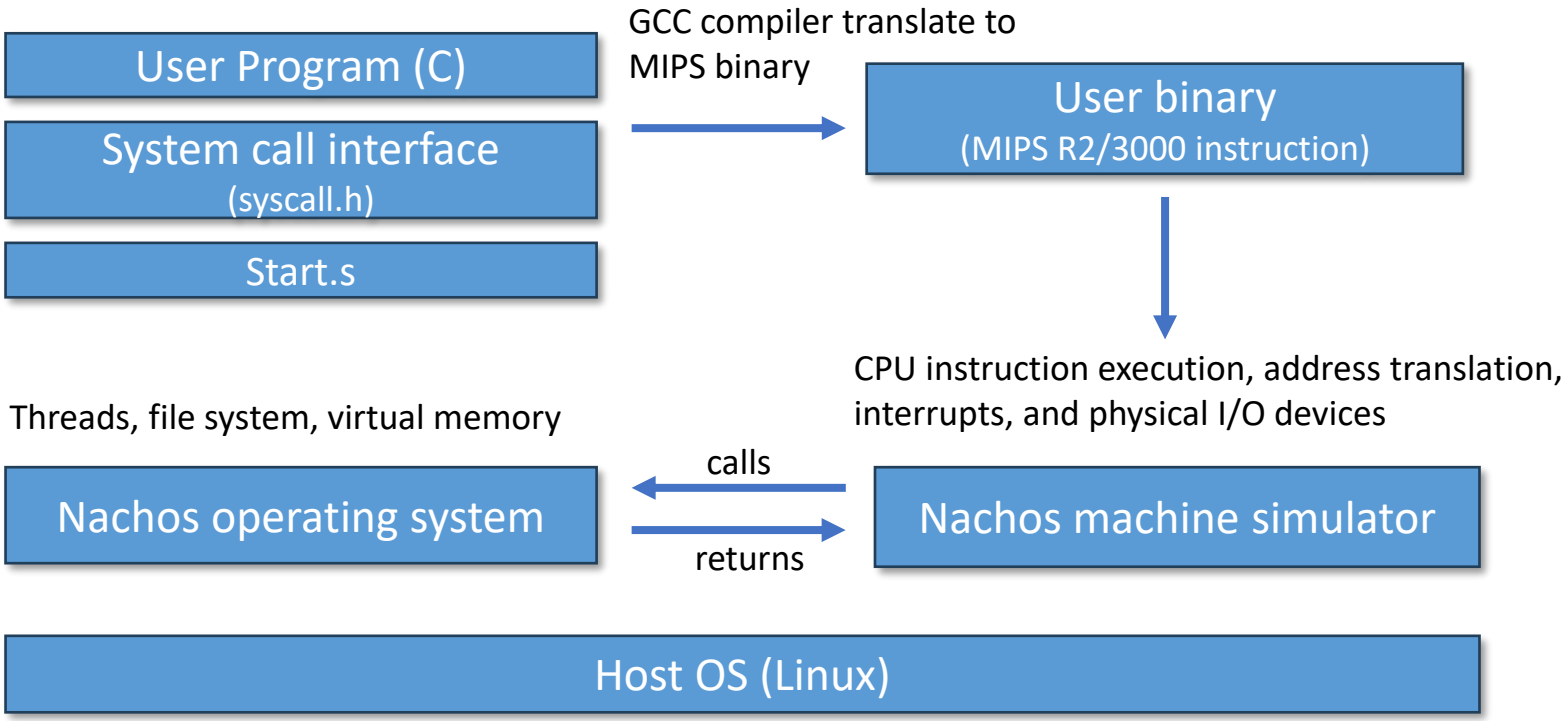
  ✓ First, is called “stub”, users can access file within nachos without own file system.

  ✓ Second, we have to implement a filesystem on top of nachos simulated disk.

- **bin/**

  - convert a normal MIPS executable into a Nachos executable (coff2noff), and others.

User Program execution flow



Nachos user program

```
13  #include "syscall.h"
14
15  int
16  main()
17  {
18      Halt();
19      /* not reached */
20  }
21
```

- code/userprog/syscall.h

  - Definitions of system call prototypes

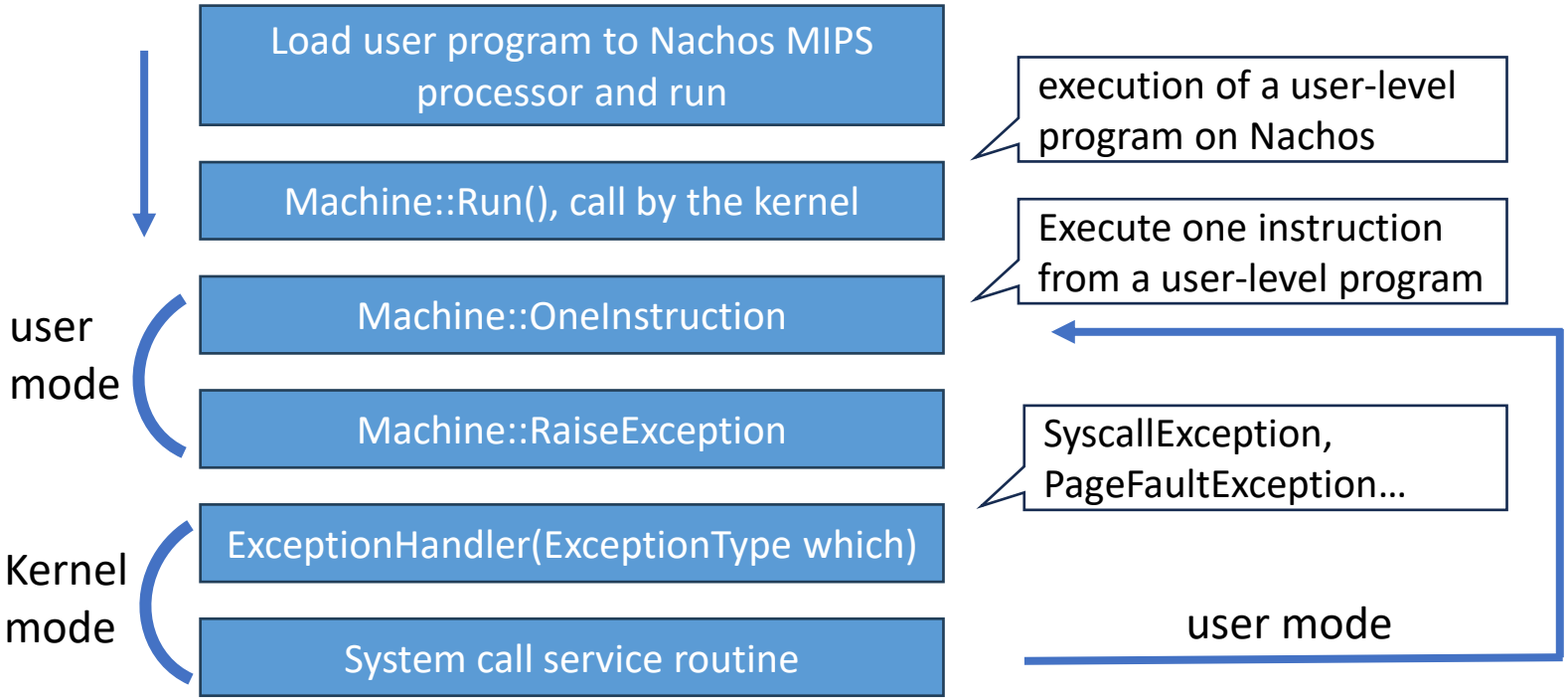
- code/test/start.S

  - Assembly language assist for user programs running on top of Nachos

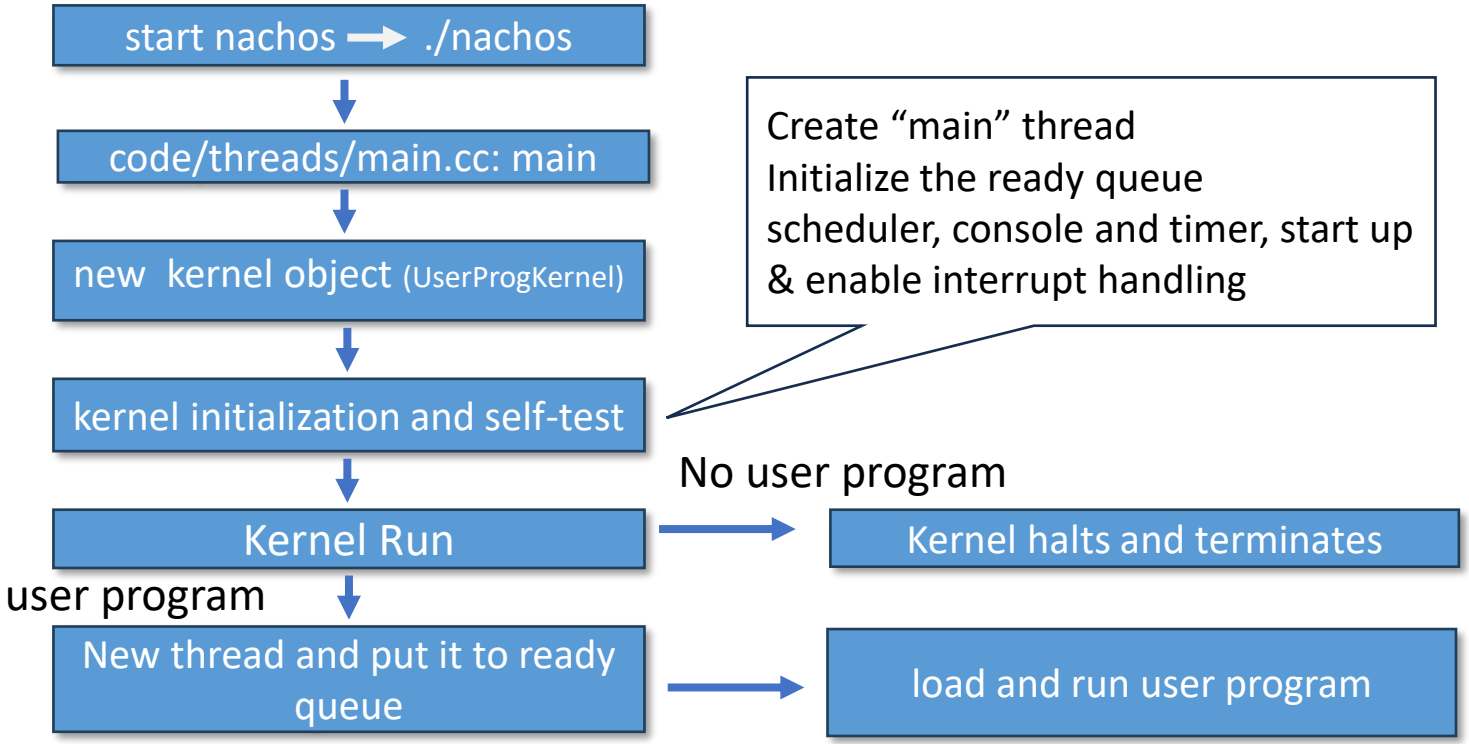
- code/userprog/exception.cc

  - syscall handler and other exceptions handling

System Call Procedure



Nachos kernel start-up flow



Run nachos

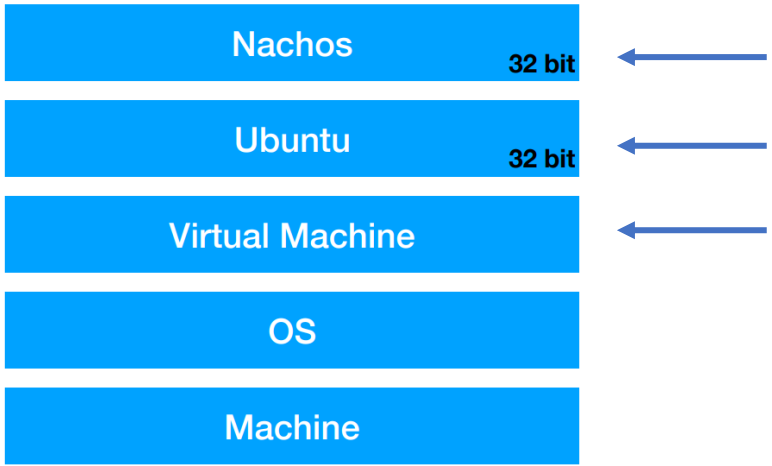
```
user@user-VirtualBox:~/Nachos/nachos-4.0/code/userprog$ ./nachos -h
argument 's' is for debugging. Machine status  will be printed
argument 'e' is for execting file.
atgument 'u' will print all argument usage.
For example:
        ./nachos -s : Print machine status during the machine is on.
        ./nachos -e file1 -e file2 : executing file1 and file2.
```

-s

Installation Guide

1. Install Virtualbox and Ubuntu 16.04 32bit
2. Please git clone the repo & follow the instruction

   https://github.com/Xueyi-Chen-David/Nachos/tree/main



Installation Guide

```
user@user-VirtualBox:~/Nachos/nachos-4.0/code/userprog$ ./nachos -e ../test/test1
Total threads number is 1
Thread ../test/test1 is executing.
Print integer:9
Print integer:8
Print integer:7
Print integer:6
return value:0
No threads ready or runnable, and no pending interrupts.
Assuming the program completed.
Machine halting!

Ticks: total 200, idle 66, system 40, user 94
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0
```

Project 1 – part1

Multi-Programming

Multi-Programming

What is multi-programing?

- The ability of an operating system executes more than one program using a single processor

Goal:

1. Implement page table mechanism

Multi-Programming

- cd ./userprog
- ./nachos –e ../test/test1
  - Print integer:9
  - Print integer:8
  - Print integer:7
  - Print integer:6
- ./nachos –e ../test/test2
  - Print integer:20
  - Print integer:21
  - Print integer:22
  - Print integer:23
  - Print integer:24
  - Print integer:25
- The programs execute correctly

Multi-Programming

- ./nachos –e ../test/test1 –e ../test/test2
  - Print integer:9
  - Print integer:8
  - Print integer:7
  - Print integer:20
  - Print integer:21
  - Print integer:22
  - Print integer:23
  - Print integer:24
  - Print integer:6
  - Print integer:7
  - …
- The result is wrong. And we are going to fix it.

Trace Code and Fix the Issue

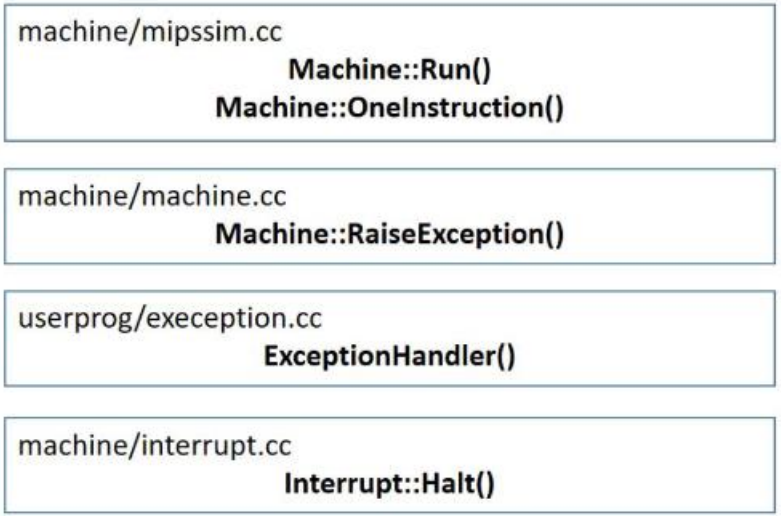
- Trace the following files and find out why the result is wrong
  - nachos-4.0/code/userprog/addrspace.h
  - **nachos-4.0/code/userprog/addrspace.cc**
  - nachos-4.0/code/userprog/userkernel.cc
  - nachos-4.0/code/machine/translate.h
  - nachos-4.0/code/machine/translate.cc
- After you fix the bug, recompile Nachos and see if the result is correct

Project 1 – part2

system call tracing

System call tracing

- Goal: Trace the SC_Halt to understand what a system call is implemented. (test/halt.c). Trace code from source file.



Dynamic debug

- ./nachos -d <debugflag>
- Gdb debug
- Use gdb-peda :
  - sudo apt install git
  - git clone https://github.com/scwuaptx/peda.git ~/peda
  - echo "source ~/peda/peda.py" >> ~/.gdbinit
  - cp ~/peda/.inputrc ~/

Debugflags: lib/debug.h

Dynamic debug

- For example: you want to see the state when execute *Interrupt::Halt*
- Set the breakpoint:

```
user@user-VirtualBox:~/Nachos/nachos-4.0/code/userprog$ gdb ./nachos -q
Reading symbols from ./nachos...done.
gdb-peda$ b Interrupt::Halt
Breakpoint 1 at 0x804d3a5: file ../machine/interrupt.cc, line 236.
```

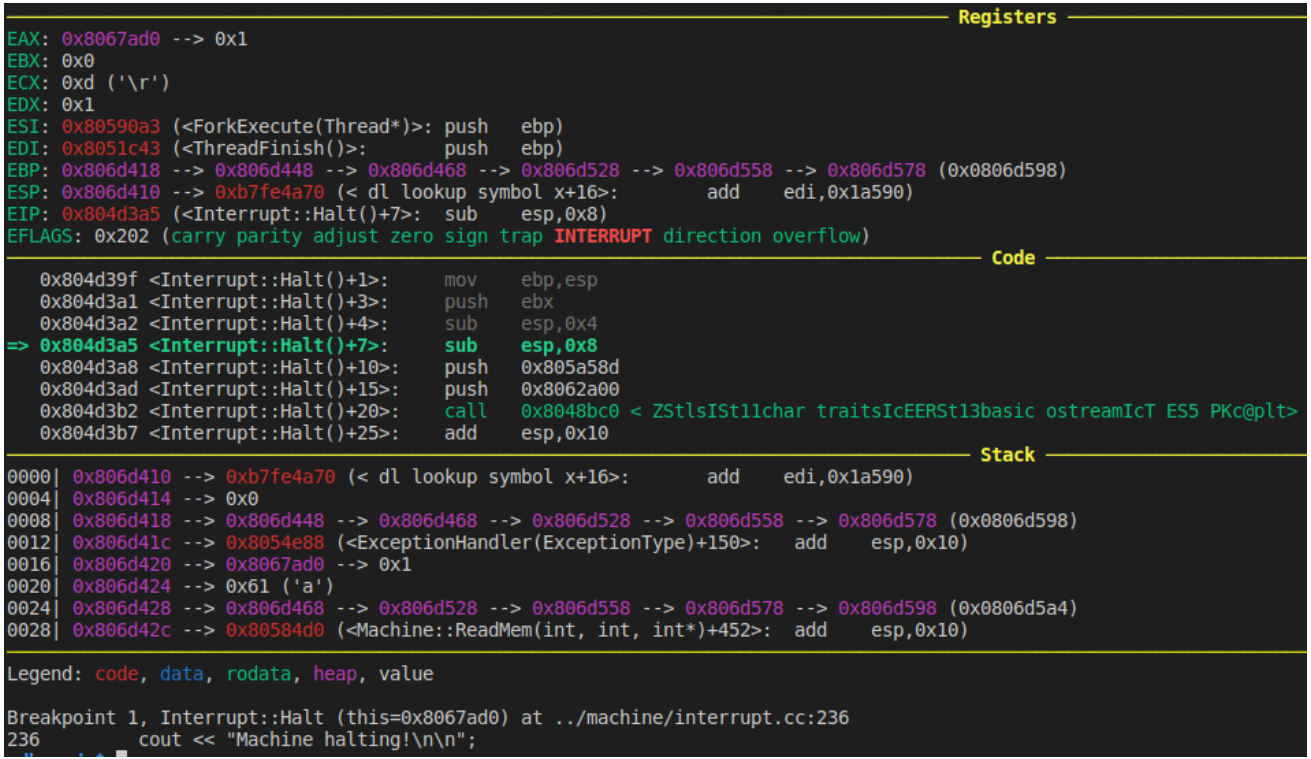
- run ../test/halt:

```
gdb-peda$ r -e ../test/halt
```

- And then… see the next page

Dynamic debug

- If gdb-peda



Dynamic debug

- Show the breakpoint info (breakpoint at 186 line)

```
gdb-peda$ info b
Num     Type           Disp Enb Address    What
1       breakpoint     keep y   0x0804d3a5 in Interrupt::Halt() at ../machine/interrupt.cc:236
        breakpoint already hit 1 time
```

- List the source code

```
gdb-peda$ list 234, 240
234     Interrupt::Halt()
235     {
236         cout << "Machine halting!\n\n";
237         kernel->stats->Print();
238         delete kernel;      // Never returns.
239     }
240
```

Dynamic debug

- ni: next instruction, si: step into function, c: continue execution
- Show the current state of call stack.

```
gdb-peda$ bt
#0  Interrupt::Halt (this=0x8067ad0) at ../machine/interrupt.cc:236
#1  0x08054e88 in ExceptionHandler (which=SyscallException) at ../userprog/exception.cc:62
#2  0x08055cb7 in Machine::RaiseException (this=0x8067d58, which=SyscallException, badVAddr=0x0) at ../machine/machine.cc:109
#3  0x08057e1a in Machine::OneInstruction (this=0x8067d58, instr=0x806d5c0) at ../machine/mipssim.cc:558
#4  0x0805632c in Machine::Run (this=0x8067d58) at ../machine/mipssim.cc:62
#5  0x08054b56 in AddrSpace::Execute (this=0x8069420, fileName=0xbffff1fe "../test/halt") at ../userprog/addrspace.cc:165
#6  0x080590cc in ForkExecute (t=0x8069238) at ../userprog/userkernel.cc:88
#7  0x0805a12e in ThreadRoot ()
#8  0x0805a126 in ?? ()
```

Report

- Report – part 1 (70%)
  - Why the result is not congruent with the expected
  - How you really modified Nachos, including some important code and descriptions
  - Experiment results and some analysis
- Report – part 2 (30%)
  - Please explain the details of each function call as much as you can

Policy

- Please save as [Student ID]_project1.pdf
  - E.g. f10921a18_project1.pdf
- Upload to NTU cool. DDL: 2023/10/19
- Penalty: decrease 5% per day

TAs

- 吳吉加 r12921093@ntu.edu.tw
- 楊冠彥 r11921091@ntu.edu.tw
- 陳學義 f10921a18@ntu.edu.tw

Reference

- A Road Map Through Nachos
- 在Windows 10中使用WSL2安装NachOS
- Vscode
- Nachos 中文教程
- Nachos Tutorial

Thanks for your attention !