國立陽明交通大學
NATIONAL YANG MING CHIAO TUNG UNIVERSITY

Institute of Artificial Intelligence Innovation
Department of Computer Science

Operating System

Homework 01: System Call

Shuo-Han Chen (陳碩漢),
shch@nycu.edu.tw

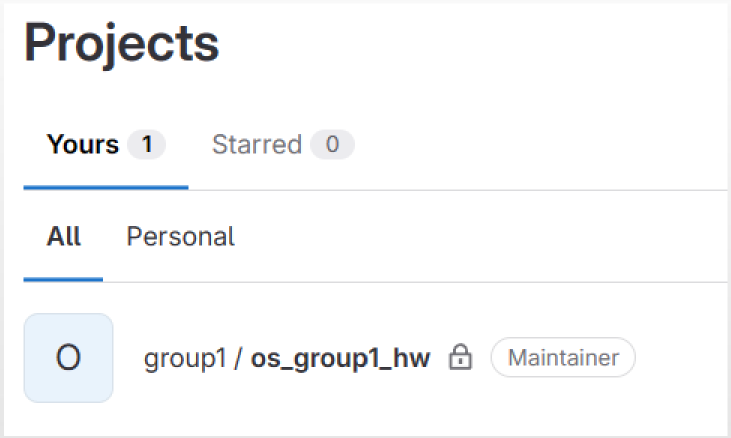
Wed. 10:10 - 12:00 EC115 +
Fri. 11:10 – 12:00 Online

Goal

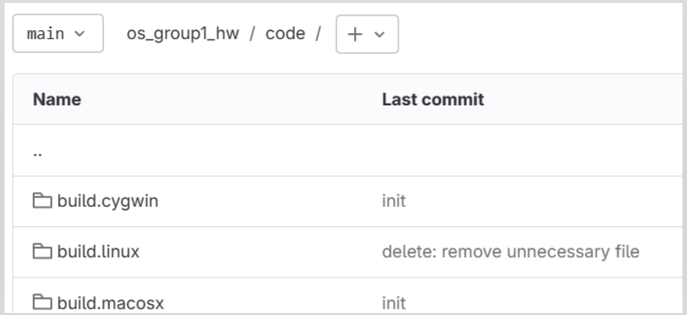
- Understand how to work in Linux Environment
- Understand how system calls are implemented by OS
- Understand the difference between user mode and kernel mode

Repository Password

- Gitlab Link: https://css-nachos.hopto.org/gitlab/
  - Account : studentID
  - Password : 013kd9123d
  - You should modify your default password
- After logging into your Gitlab account, you should see your group project
- Your Nachos file will already be inside the project



- Jenkins Link: https://css-nachos.hopto.org/jenkins/
  - Account : studentID
  - Password : 013kd9123d
  - You should modify your default password



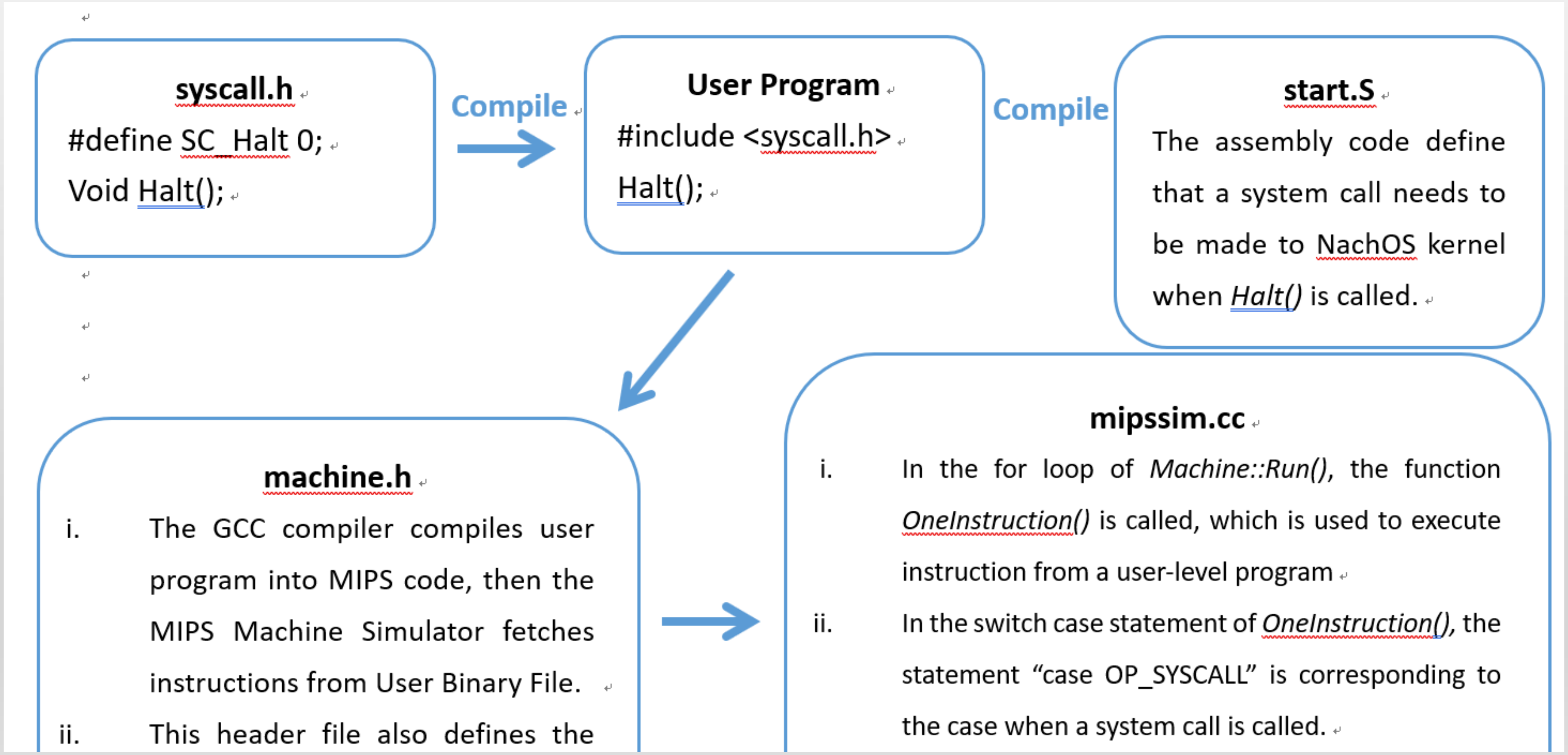
Name	Last commit
..	
build.cygwin	init
build.linux	delete: remove unnecessary file
build.macosx	init

Part I

- Trace how Halt() system call works
    - Explain how system calls go through NachOS in details
- Trace how Create() system call works
    - Explain the basic operations and data structure in a file system
- Trace the Makefile in code/test/Makefile to understand how test files are compiled
- Files to look into
    - userprog/syscall.h, exception.cc, ksyscall.h, synchconsole, console
    - machine/mipssim, interrupt
    - filesys/openfile, filesys
    - test/start.s, halt.c, Makefile
    - threads/kernel
- You should include two things in the report
    - Flow chart of system call (Halt, Create)
    - Tracing details of code (Halt, Create, Makefile)

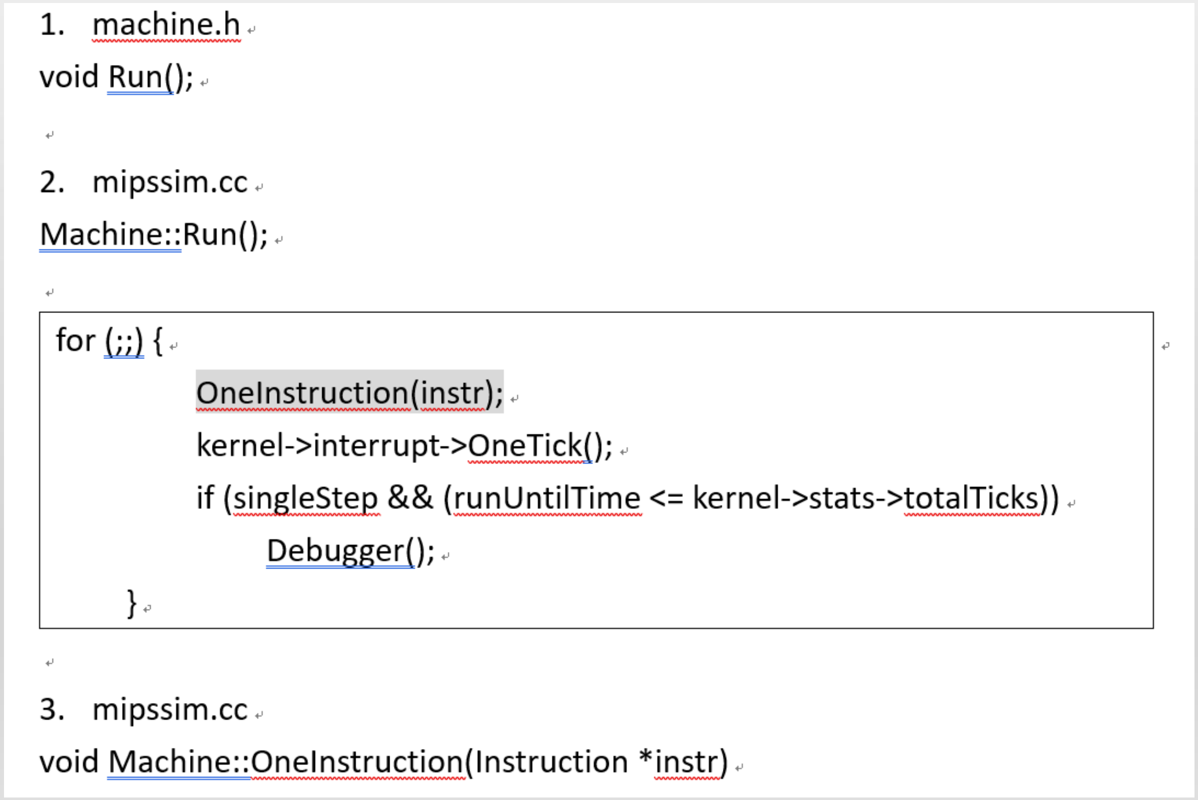
Flow Chart of System Call

- It should look like …


syscall.h
#define SC_Halt 0;
Void Halt();
Compile
User Program
#include <syscall.h>
Halt();
Compile
start.S
The assembly code define that a system call needs to be made to NachOS kernel when *Halt()* is called.
machine.h
i. The GCC compiler compiles user program into MIPS code, then the MIPS Machine Simulator fetches instructions from User Binary File.
ii. This header file also defines the
mipssim.cc
i. In the for loop of *Machine::Run()*, the function *OneInstruction()* is called, which is used to execute instruction from a user-level program
ii. In the switch case statement of *OneInstruction()*, the statement "case OP_SYSCALL" is corresponding to the case when a system call is called.

Tracing Details of Code

- Just paste the code with nice arrangement
- Don't paste the whole file, just the part that will be used



1. machine.h

```
void Run();
```

2. mipssim.cc

```
Machine::Run();
```

```
for (;;) {
        OneInstruction(instr);
        kernel->interrupt->OneTick();
        if (singleStep && (runUntilTime <= kernel->stats->totalTicks))
            Debugger();
    }
```

3. mipssim.cc

```
void Machine::OneInstruction(Instruction *instr)
```

Part II

- Implement a console I/O system call

```
void PrintInt (int number)
        // Output the number and a line separator to the console.
```

- Implement four file I/O system call

```
OpenFileId Open(char *name);
        // Open a file with the name, and returns its corresponding OpenFileId.
        // Return -1 if open fails
int Write(char *buffer, int size, OpenFileId id);
        // Write “size” characters from buffer into the file
        // Returns number of characters actually written to the file
        // If attempt writing to an invalid id, return -1
int Read(char *buffer, int size, OpenFileId id);
        // Read “size” characters from file into the buffer
        // Returns number of characters actually read from the file
        // If attempt reading from an invalid id, return -1
int Close(OpenFileId id);
        // Close the file with id
        // Return 1 if successfully close the file, 0 otherwise
```

Requirement

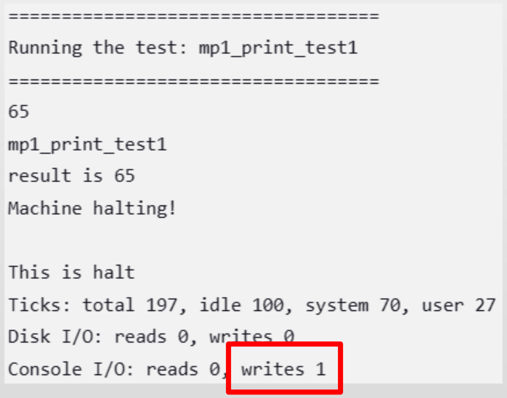
- All your implemens should not use any IO functions from standard libraries (e.g. printf(), cout, fopen(), fwrite(), write(), etc.).
- Must handle invalid file open requests, including the non-existent file, exceeding opened file limit (at most 20 files)
- Must handle invalid file read, write, close requests, including invalid id

Hint

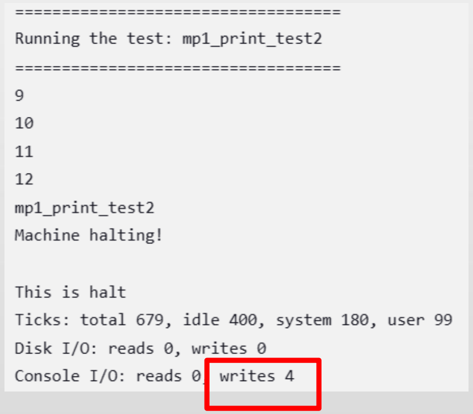
- We use the stub file system for this homework, so Do not change or remove the flag -DFILESYS_STUB in the Makefile under build.linux/
- You can run consoleIO_test1.c, consoleIO_test2.c to verify consoleIO part
- You can run fileIO_test1.c, fileIO_test2.c to verify fileIO part

Jenkins os_group_ta Description

- You will have 8 test cases, and each test case is 9% of the total grade
- In print test, you should verify that the number of Console I/O writes is correct

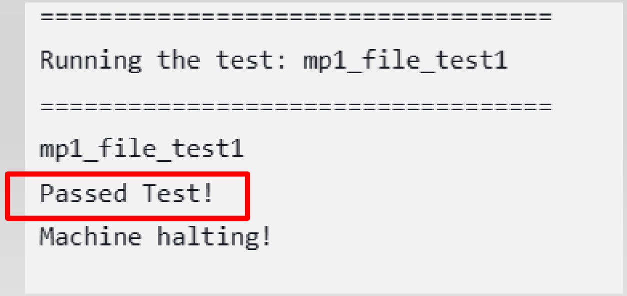


```
===================================
Running the test: mp1_print_test1
===================================
65
mp1_print_test1
result is 65
Machine halting!

This is halt
Ticks: total 197, idle 100, system 70, user 27
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 1
```



```
===================================
Running the test: mp1_print_test2
===================================
9
10
11
12
mp1_print_test2
Machine halting!

This is halt
Ticks: total 679, idle 400, system 180, user 99
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 4
```

- In the file test, you should verify that your output includes the string "Passed Test!" for each test



```
===================================
Running the test: mp1_file_test1
===================================
mp1_file_test1
Passed Test!
Machine halting!
```

Grading

- PartI (Trace System call) - 25%
- PartII (Implement System call)- 72%
    - Console I/O system call - 24%
    - File I/O system call - 48%
- Report Format - 3%

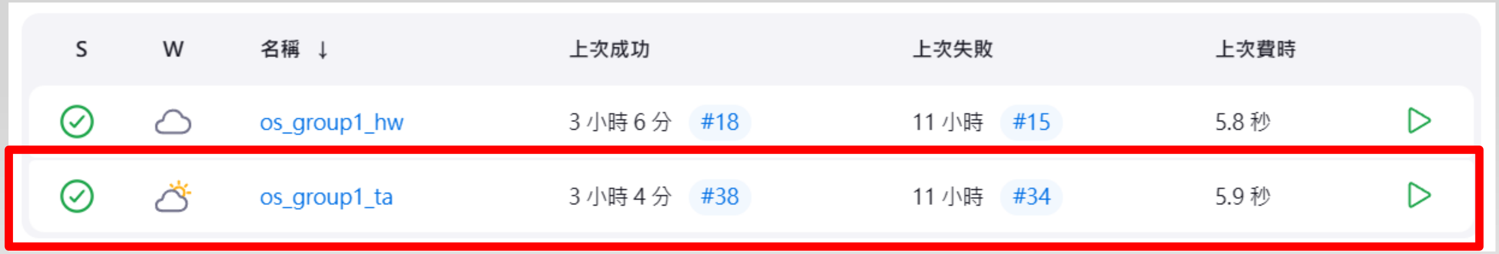
- Deadline: 10/5 23:59

Report Format

- Please follow the word file to form your report for HW01
- Format guide
  - Content format: should be set with 12pt front,16pt row height, and align to the left.
  - Caption format: 18pt and Bold font.
  - Font format: Times New Roman.
  - Figure: center with single line row height.
  - Change the title to your student ID and name in Chinese.
  - Upload pdf file with the file name format : OS_HW01_GROUP_X.pdf (change X to your group ID)

Reminder

- The homework is considered passed **only if the TA job** passes
- Feel free to ask TA questions
  - The TA will only assist you with GitLab, Jenkins environment problems, or any issues related to homework requirements.
  - The TA will not help you debug your code.
  - Teams Message(Recommended): 簡子茸、徐翊安
  - Email:
    - tzerongjian.cs13@nycu.edu.tw
    - vm6u40.cs13@nycu.edu.tw



S	W	名稱 ↓	上次成功	上次失敗	上次費時	
✓	☁	os_group1_hw	3 小時 6 分 #18	11 小時 #15	5.8 秒	▷
✓	⛅	os_group1_ta	3 小時 4 分 #38	11 小時 #34	5.9 秒	▷

Q & A

Thank you for your attention