Project Overview: **PINTOS**

Instructor: Prof. Seokin Hong (seokin@knu.ac.kr)
TA : Seungmin Shin (steve6238@naver.com)

1. Description

The goal of this project assignments is to implement some features of a simple operating system. For this assignment, we will use Pintos which is a simple operating system framework for the 80x86 architecture. Pintos supports key features of operating system such as kernel threads, loading and running user programs, and a file system, but it implements all of these in a very simple way. So, your goal is to extend the features and implement a virtual memory.

There will be three projects. In the first project, you will reimplement **timer_sleep()** function that suspends execution of the calling thread until time has advanced by at least x timer ticks. In the second project, you will implement the **argument passing** that is required to run a program and also will implement **system call handler**. In the third project, you will implement paging and stack growth. These projects require substantial work; the average time to complete each project would be about 30-40 hours. Even if it'll not be easy for you to do all assignments, you will learn a lot about actual implementation of the operating system.

2. Project Schedule

Project	Topic	Sub-topic	Due Date
PJ1	Threads	Alarm clock	04/19
PJ2	User Program	Augment passing, System call	05/17
PJ3	Virtual Memory	Paging, Stack Growing	06/14

3. Pintos Resources

a. Stanford website for PintOS
b. PintOS guides and tutorials will be available in LMS

4. Submission Instructions

~~We will use a private git repository for submission. So, please be familiar with the Git. I'll give you a detailed submission instruction later.~~

Submit a compressed file that include Pintos source code to LMS [updated April 8, 2019]

- A way to compress the Pintos folder

```
seokin@compasslab1:~/test$ tar cvf team1_pj1.tar.gz pintos
pintos/
pintos/.git/
pintos/.git/refs/
pintos/.git/refs/heads/
pintos/.git/refs/heads/master
pintos/.git/refs/tags/
pintos/.git/refs/remotes/
pintos/.git/refs/remotes/origin/
```

Late Day Policy

This project is due at 11:59pm on the due date. A grading penalty will be applied to late assignments. Any assignment turned in late will be penalized 25% per late day.

Plagiarism

No plagiarism will be tolerated. If the assignment is to be worked on your own, please respect it. If the instructor determines that there are substantial similarities exceeding the likelihood of such an event, he will call the two (or more) students to explain them and possibly to take an immediate test (or assignment, at the discretion of the instructor) to determine the student's abilities related to the offending work.

Project 1: Thread

Instructor: Prof. Seokin Hong (seokin@knu.ac.kr)
TA : Seungmin Shin (steve6238@naver.com)

Assigned: April 2, 2019
Due: 11:59 pm April 19, 2019

1. Description

In this project, your job is to extend the functionality of a minimally functional thread system. You will be mainly working in the "threads" directory for this project. For more information, Please read a background section (2.1) in https://web.stanford.edu/~ouster/cgi-bin/cs140-winter13/pintos/pintos.pdf to come to know the thread implementation of PintOS.

The function you need to extend is the timer_sleep() that is defined in 'devices/timer.c'. The current version of time_sleep() function is working. However, it "busy waits", which means that it keep checking the current time and calling thread_yield(). So, you need to reimplement the timer_sleep() to avoid the "busy waiting".

void timer_sleep (int64_t ticks)

- Description
    - Suspends execution of the calling thread until time has advanced by at least x timer ticks.
    - Unless the CPU is otherwise idle, the thread need not wake up after exactly x ticks. Just put it on the ready queue after they have waited for the right amount of time.
    - The argument to timer_sleep() is expressed in timer ticks, not in milliseconds or any another unit. There are TIMER_FREQ timer ticks per second, where TIMER_FREQ is a macro defined in devices/timer.h. The default value is 100. We don't recommend changing this value, because any change is likely to cause many of the tests to fail.

2. What you need to submit

1) **Design document (see section 5 below)**

   a. You need to copy the design document in "src/threads" directory of the PintOS source code when you submit it.

2) **PintOS source code.**

3. How to evaluate your code.

a. We will evaluate your code with the following steps

```
$ cd src/threads
$ make
$ cd build
$ make check
```

b. Test results should be..

```
alarm-single PASSED
alarm-multiple PASSED
alarm-simultaneous PASSED
alarm-zero PASSED
alarm-negative PASSED
```

4. Useful Resource is available in LMS

a. PintOS installation tutorial (slides)

b. Project 1 guide (slides)

5. Documentation template

```
---- GROUP ----

>> Fill in the names and email addresses of your group members.

FirstName LastName <email@domain.example>
FirstName LastName <email@domain.example>
FirstName LastName <email@domain.example>

---- PRELIMINARIES ----

>> Describe briefly which parts of the assignment were implemented by each
member of your team and specify the contribution between your member, say 3:3:4,
or 1:3:6.

FirstName LastName: contribution
FirstName LastName: contribution
FirstName LastName: contribution

                         ALARM CLOCK
                         ===========


---- DATA STRUCTURES ----

>> A1: Copy here the declaration of each new or changed `struct' or
>> `struct' member, global or static variable, `typedef', or
>> enumeration.  Identify the purpose of each in 25 words or less.



---- ALGORITHMS ----

>> A2: Briefly describe what happens in a call to timer_sleep(),
>> including the effects of the timer interrupt handler.

>> A3: What steps are taken to minimize the amount of time spent in
>> the timer interrupt handler?
```