```
+--------------------+
|        CS 140      |
| PROJECT 1: THREADS |
|   DESIGN DOCUMENT  |
+--------------------+
```

---- GROUP ----

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

Junjie Wei <junjiewe@buffalo.edu>
FirstName LastName <email@domain.example>
FirstName LastName <email@domain.example>

---- PRELIMINARIES ----

>> If you have any preliminary comments on your submission, notes for the
>> TAs, or extra credit, please give them here.

Note: none of my team mate respond me when i contact them, so i have to do it all by myself, i don't even know how they look like......

>> Please cite any offline or online sources you consulted while
>> preparing your submission, other than the Pintos documentation, course
>> text, lecture notes, and course staff.

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.

```
struct timer_sleep_for_thread{
        struct* thread;
        int64_t wake_up;
        struct list_elem slp_elem;
};
```

struct thread - is the current thread that will be sleeping.
wake_time- the purpose for this is that it will help to identify the time when will the thread wake up while timer ticks and match the wake up time.

struct list_elem slp_elem - the list of element for the sleeping threads list.

---- ALGORITHMS ----

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

When the timer_sleep function is called, it will first check the thread is match the tick value, if it matches, then it will be put in the sleeping thread list. Then, thread block function will be called to block the thread from keep on running. When the ticks value does not match the global tick value, then it will be unblocked.

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

since we keep the sleeping thread in the sleeping list, and the list will keep track of the order to let the interrupt handler to use, which is the first on the list. Therefore it will be more time efficient for the timer interrupt handler to operates.

---- SYNCHRONIZATION ----

>> A4: How are race conditions avoided when multiple threads call
>> timer_sleep() simultaneously?

when the threads call timer_sleep simultaneously, since interruption is turned off when list operates. So, only one threads is allow to sleep at that time period.

>> A5: How are race conditions avoided when a timer interrupt occurs
>> during a call to timer_sleep()?

The interrupt function is turned off.

---- RATIONALE ----

>> A6: Why did you choose this design? In what ways is it superior to
>> another design you considered?

I did not consider other design since the design above is reasonable.

PRIORITY SCHEDULING

---- DATA STRUCTURES ----

>> B1: 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.

```
struct thread {
        ..
        int priority_before;
        struct lock *blocked_lock;
        struct list locks;
}


struct lock {
        ..
        int priority;
        list_elem lock_elem;
}
```

>> B2: Explain the data structure used to track priority donation.
>> Use ASCII art to diagram a nested donation. (Alternately, submit a
>> .png file.)

---- ALGORITHMS ----

>> B3: How do you ensure that the highest priority thread waiting for
>> a lock, semaphore, or condition variable wakes up first?

when insert the element into the waiting list, i will check their priority, and put it in a data structure such as a priority queue, so whenever the resources is available for threads, the first element will always got pop out from the queue since it always has the higher priority.

>> B4: Describe the sequence of events when a call to lock_acquire()
>> causes a priority donation. How is nested donation handled?

The current thread calls lock_acquire, it will wait on the lock first. It will then check the next thread's priority, so the and if the value its the same then priority donation will start. the next thread will donate one priority to the current thread, so on and so forth. Then the current thread will take the lock and execute. After, it finishes, it will release the lock and the next thread with higher priority will take the lock.

>> B5: Describe the sequence of events when lock_release() is called
>> on a lock that a higher-priority thread is waiting for.

Set the lock_holder equals to null, since that thread finished execute and does not need the lock. Then the next thread that has higher priority will be chosen to acquire the lock to execute. Then that thread will remove the lock from the lock list.The lock list will check wether if that thread can still donate it's priority to other threads.

---- SYNCHRONIZATION ----

>> B6: Describe a potential race in thread_set_priority() and explain
>> how your implementation avoids it. Can you use a lock to avoid
>> this race?

The potential race can be happen in the situation when two threads are trying to set eachother's priority at the same time, then the order of executing the threads will be messed up. We can use the lock to avoid the race by lock each thread before they called thread_set_priority.

---- RATIONALE ----
>> B7: Why did you choose this design? In what ways is it superior to
>> another design you considered?

This design is more safe then the other design, becaue it can avoid potential race conditions.

ADVANCED SCHEDULER

---- DATA STRUCTURES ----

>> C1: 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.
thread struct {
        ..
        int nice;
        int recent_cpu;
}

In thread.c add the following:
        int load_avg;

---- ALGORITHMS ----

>> C2: Suppose threads A, B, and C have nice values 0, 1, and 2. Each

>> has a recent_cpu value of 0. Fill in the table below showing the
>> scheduling decision and the priority and recent_cpu values for each
>> thread after each given number of timer ticks:

timer ticks	recent_cpu A	B	C	priority A	B	C	thread to run
0	0	1	2	63	61	59	A
4	4	0	0	62	61	59	A
8	8	0	0	61	61	59	B
12	8	4	0	61	60	59	A
16	12	4	0	60	60	59	B
20	12	8	0	60	59	59	A
24	16	8	0	59	59	59	B
28	16	12	0	59	58	59	C
32	16	12	4	59	58	58	A
36	20	12	4	58	58	58	B

>> C3: Did any ambiguities in the scheduler specification make values
>> in the table uncertain? If so, what rule did you use to resolve
>> them? Does this match the behavior of your scheduler?
The ambiguities is the priority. I will choose the one that it is in
front of the wait list and donate the priority of the next thread to
the first to let it execute first.

>> C4: How is the way you divided the cost of scheduling between code
>> inside and outside interrupt context likely to affect performance?

---- RATIONALE ----

>> C5: Briefly critique your design, pointing out advantages and
>> disadvantages in your design choices. If you were to have extra
>> time to work on this part of the project, how might you choose to
>> refine or improve your design?

>> C6: The assignment explains arithmetic for fixed-point math in
>> detail, but it leaves it open to you to implement it. Why did you
>> decide to implement it the way you did? If you created an
>> abstraction layer for fixed-point math, that is, an abstract data
>> type and/or a set of functions or macros to manipulate fixed-point
>> numbers, why did you do so? If not, why not?

SURVEY QUESTIONS

Answering these questions is optional, but it will help us improve the

course in future quarters. Feel free to tell us anything you
want--these questions are just to spur your thoughts. You may also
choose to respond anonymously in the course evaluations at the end of
the quarter.

>> In your opinion, was this assignment, or any one of the three
problems
>> in it, too easy or too hard? Did it take too long or too little
time?

It was ok, it took me decent amount of time to understand and answer
the questions.

>> Did you find that working on a particular part of the assignment
gave
>> you greater insight into some aspect of OS design?

Every part that i've been working on help me understand better in the
OS design.

>> Is there some particular fact or hint we should give students in
>> future quarters to help them solve the problems? Conversely, did
you
>> find any of our guidance to be misleading?

Nope, everything was pretty clear.

>> Do you have any suggestions for the TAs to more effectively assist
>> students, either for future quarters or the remaining projects?

No.

>> Any other comments?

No.