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

---- GROUP 33----

Aayush Prasad 18CS30002 <aayuprasad@gmail.com>
Rajdeep Das 18CS30034 <rajdeepdasren@gmail.com>

---- PRELIMINARIES ----

The shutdown port has changed in the latest version of qemu. Thus we added a new line in /src/devices/shutdown.c “outw (0xB004, 0x2000)”

“SIMULATOR = --qemu” is changed to “SIMULATOR = --qemu” in /pintos/src/threads/Make.vars

In pintos and pintos gdb, $HOME is changed to /home/aayush because PERL does not support using $HOME.

Reference: https://stackoverflow.com/a/45276093

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.

1. In pintos/src/threads/threads.c:
    a. struct list mlfqs_list[PRI_MAX + 1]: mlfqs list array containing threads in the priority range PRI_MIN (0) to PRI_MAX (63)
    b. fixed_point_t load_avg: load average value which is calculated using exponentially weighted moving average method
    c. int ready_threads: number of ready threads
    d. #define put_in_mlfqs(T) list_push_back(&mlfqs_list[T->priority], &T->elem): shortcut to put threads in appropriate mlfqs array
    e. thread_int(), thread_tick(), thread_block(), thread_unblock(), thread_exit(), thread_yield(), thread_set_priority(), thread_get_priority(), thread_set_nice(), thread_get_nice(), thread_load_avg(), thread_get_recent_cpu(), init_thread(), next_thread_to_run(), comp() , thread_cmp_fnc(),
2. In pintos/src/threads/thread.h:
    a. threads/synch.h and threads/fixed-point.h were included
    b. fixed_point_t nice: thread nice value

c. fixed_point_t recent_cpu: thread recent cpu value
d. thread_cmp_fnc(): comparator function on threads based on priority
e. comp(): utility function for thread_cmp_fnc()

3. In pintos/src/devices/timer.c:
    a. struct list sleep_list: for sleeping threads
    b. struct lock sleep_list_lock: locks sleep_list for synchronization
    c. timer_init(), cmp_fnc(), timer_sleep(), timer_wakeup(),
4. In pintos/src/devices/timer.h:
    a. struct sleep_list_elem: containing the following data members:
        i. struct list_elem list_el: for inserting into list
        ii. struct semaphore* sem: Semaphore for unblocking sleeping threads
        iii. struct thread* curr_thread: sleeping thread
        iv. Int64_t tick: Predetermined time for wakeup (wakeup_time)

---- 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			priority			thread to run
	A	B	C	A	B	C	
0	0	0	0	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	C
28	16	8	4	59	59	58	B
32	16	12	4	59	58	58	A
36	20	12	4	58	58	58	C

>> 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?

A: When recent_cpu is calculated, the time that CPU spends on the calculations every 4 ticks for e.g. load_avg, recent_cpu for all threads, priority for all threads. When the CPU

does these calculations, the current thread cannot run. Thus, every 4 ticks, the real ticks added to recent_cpu is less than 4 ticks. However, here 4 ticks were added.

>> C4: How is the way you divided the cost of scheduling between code
>> inside and outside interrupt context likely to affect performance?
If the CPU spends too much time on calculations, it takes away most of the time for the thread to execute. This will raise its load_avg, recent_cpu and therefore lower its priority because it does not get enough time in 4 ticks. So, if the cost of scheduling in the interrupt context goes up, the performance will be lowered.

---- 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?
A: Since there are a constant number of priorities, creating an array of 64 queues and iterating through them from highest to lowest priority is an efficient way to find the highest priority ready/running thread regardless of the number of threads. Also, it does not take up too much memory because 64 linked lists uses roughly the same amount of memory as 1 linked list with the same number of elements.

>> 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?
A: We created an abstract data type and set of functions for fixed-point arithmetic. This is done because pintOS has disabled float numbers. Instead of float-numbers, we have used fixed-pointed numbers to represent recent_cpu and load_avg. The data type and the set of functions was included in a file named fixed-point.h in the threads folder.

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?

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

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

>> Any other comments?