Assignment T01

Data Structure:

Code:

```
In thread.h:
struct thread
 {
   int64_t wakeup_at;                   /* Time at which sleeping thread must be
woken up */
   int original_priority;               /* An int variable to save previous
priority */
   struct list_elem sleepers_elem;      /* Sleepers List element */
 };

In thread.c:
static int64_t next_wakeup_time;
static struct list sleepers;
```

Algorithm:

In timer.c, timer sleep function was changed according to the following code:

```
void
timer_sleep (int64_t ticks)
{
 int64_t start = timer_ticks ();
 int64_t wake_time = start + ticks;
 ASSERT (intr_get_level () == INTR_ON);
 // while (timer_elapsed (start) < ticks)
 //   thread_yield ();

 if (ticks <= 0)
   return;
 // New Algorithm implemented.
 thread_set_priority_temporarily_up();
 thread_block_till(wake_time);
 thread_set_next_wakeup();
 thread_priority_restore();
 }
```

Four new functions were introduced in thread.c, namely :

```
void thread_set_priority_temporarily_up(void);
void thread_priority_restore(void);
void thread_block_till(int64_t);
void thread_set_next_wakeup(void);
```

Explanation:

`void thread_set_priority_temporarily_up(void):`
This function will set the priority of the current thread as PRI_MAX, and then it will store its original priority in original_priority variable.

`void thread_priority_restore(void):`
This function will restore thread’s priority from its original_priority variable.

`void thread_block_till(int64_t):`
This function will block the current thread till given input time (i.e. block the thread and set its wakeup_at variable given to the input time).

`void thread_set_next_wakeup(void):`
This function will set the value of next_wakeup_time global variable (defined in thread.c) acc. to the minimum wake_up time in sleepers list.

Finally a while loop was added in thread_ticks function, which will wake up a thread in sleepers list, if the current time becomes greater than or equal to next_wakeup_time and then will set next_wakeup_time by calling thread_set_next_wakeup function.

Synchronization:

Synchronization has to be taken care of while blocking the thread i.e in thread_block_till function. For that we used interrupts. Before blocking and inserting the thread in the sleepers list we are disabling the interrupt. After thread blocking is done, we will again restore previous interrupt level.

Rationale:

The idea was to change busy waits in timer_sleep function so as to increase CPU utilization. For that we thought we blocking the thread and introduced a list of blocked threads called sleepers.
Now why we set the priority of current thread in timer_sleep function as PRI_MAX because we want the thread to sleep, and this process should not be preempted by any other thread. That's why priority was set to max. Then thread_block till is called to block and insert the thread in sleepers list. Now next_wakeup_time global variable value is set in the function. And finally priority of blocked thread is restored.
The blocked thread is unblocked whenever current time becomes greater than equal to its wakeup time.
That's all.