ICT1012

OPERATING SYSTEMS

LABORATORY INSTRUCTIONS

Quiz 2 – Question 3

Contents

Lab challenge task (7 marks)

Prerequisite:

Download the "**quiz2-q3.zip**" file from "LMS/xSiTe" folder:

"Lab Quizzes (Prerequisites) / Quiz 2 / Question 3"

Task: xv6 user thread (user threads can go to sleep)

Context

1. You are given the xv6 operating system with an implementation of a user threads library in the files *"uthread.c", "uthread.h"* and *"uthread_switch.s"*. This implementation only consider threads that are under execution (RUNNING state) and waiting to be executed (RUNNABLE state).
2. Files *"uthread_ex0.c", "uthread_ex1.c", "uthread_ex2.c" and "uthread_ex3.c"* import the user thread library to implement and execute three user threads.
3. Even if you do not implement the required functionality, the code will compile correctly. However it will not pass the test cases.

Task Requirements

1. Thread sleep and Wake Up

We want to extend the thread functionality allowing threads to go to sleep by calling a function ***void thread_sleep()*** and to be waken up by another thread by invoking a function ***int thread_wakeup(int thread_id)***.

After invoking ***thread_sleep()*** the current thread will go to sleep and another thread will be executed.

After invoking ***thread_wakeup(thread_id)*** the thread corresponding to *thread_id* will be awaken if and only if it was sleeping and the input parameter ***thread_id*** is a valid thread index. After that the caller will continue its execution, i.e., no other thread is scheduled. This function will return 0 if succeeds and -1 if it does not.

Implement ***void thread_sleep()*** and ***void thread_wakeup(int thread_id)*** accordingly to the description above in *"uthread.c". "uthread.c"* **provides empty shells for these two functions**. Complete the provided functions with the necessary code.

2. Wake up sleeping thread when there are no active threads

It might be the case that **there are no runnable threads** when the schedule is searching threads to be executed, but there are threads in a sleeping state.

In that case, the scheduler will awake all the threads in a sleeping state, and it will set to run the thread with the smallest thread_id.

To implement such functionality, you must modify the function ***void thread_schedule(void)***. The provided code contains placeholders in key places that you have to substitute with the right code.

Task Testing

1. The user library and the programs using it run in xv6.

   Executing "***uthread_ex0***" and "***uthread_ex1***" in "*qemu*" tests "*thread_sleep*" and "*wake_up*" functionality:

```
$ make qemu
$ uthread_ex0
thread_a started
thread_b started
thread_c started
thread_a 0
thread_a 1
thread_a 2
thread_a: exit after 3
thread_b 0
thread_b 1
thread_b 2
thread_b: exit after 3
thread_c 0
thread_c 1
thread_c 2
thread_c: exit after 3
thread_schedule: no runnable threads
```

2. Executing "***uthread_ex2***" and "***uthread_ex3***" in "*qemu*" additionally tests the functionality to wake up sleeping threads when the scheduler cannot find any runnable thread:

```
$ make qemu
$ uthread_ex2
thread_a started
thread_b started
thread_c started
thread_a 0
thread_a 1
```

```
thread_a 2
thread_a: exit after 3
thread_b 0
thread_b 1
thread_b 2
thread_b: exit after 3
thread_c 0
thread_c 1
thread_c 2
only thread sleepings, wake them all
thread_c: exit after 3
thread_schedule: no runnable threads
```

3. Testing with the grading script "**grade-quiz**":

```
$ cd ..
$ ./grade-quiz uthread
== Test uthreads == uthreads: OK (6.5s)
$
```

4. Testing with the "**make grade**"

```
$ make clean && make grade
== Test uthread ==
$ make qemu-gdb
uthread: OK (6.7s)
Score: 7/7
$
```

NOTE: The grade script does not cover all the possibilities and it is your responsibility to carry out further testing

Quiz Submission

1. After "***make grade***" and "***make zipball***", submit the "***lab.zip***" file to Gradescope Assignment "***xv6labs-quiz2-q3-uthread***" for autograding.

   Note: Create the zip file with "***make zipball***" and not with Windows zip program or iOS Files App to avoid zipping multiple hierarchical directories.
2. Autograder will execute the same "*make grade*" when you submit your "*lab.zip*" file. This is for submission verification only. The score will not be included in the total score.
3. Test case script may not cover all the possibilities and it is your responsibility to carry out further testing,
4. After the quiz due date and time, all submissions will be regraded again with different set of test cases.

5. To ensure your submission goes through without technical issues, please submit at the earliest.