Nachos Assignment 2: Synchronization

Sanzheng Qiao
Department of Computing and Software

February 27, 2013

Due: March 11, Monday, 23:59:59.

In this assignment you continue to modify and extend the existing thread system and solve several synchronization problems.

1. (32 marks) Implement `void Thread::Join()` in the Thread class. Add an argument to the thread constructor that says whether or not a Join will be called on this thread. The following code segment illustrates how this function is used:

   ```
   t = new Thread("forked thread", TRUE);
   t->Fork(function, arg);
   t->Join();
   ```

   The second argument in the thread constructor indicates that a Join may be called on the forked thread `t`. In the above code, Join is indeed called. The thread calling Join is blocked until the forked thread `t` finishes. Your solution should work when Join is not called. In particular, your solution should properly delete the thread control block whether or not Join is to be called, and whether or not the forked thread finishes before the Join is called.

2. (32 marks) Complete the implementation of the "alarm clock" class in threads/alarm by implementing

   `WaitUntil(int x)`.

   Threads call `WaitUntil(x)` to suspend execution until time has advances to at least `x` from now. This is useful for threads that operate in real-time, for example, for blinking the cursor once per second. There is no requirement that threads start running immediately after waking up; just put them on the ready queue after they have waited for approximately the right amount of time.