Assignment 2

Sanzheng Qiao

Department of Computing and Software

March, 2013
Addition to Thread structure (thread control block):

- joinable, boolean
- list, joinableList
- Semaphore *joinSemaphore, constructed with initial value 0 if joinable
- Semaphore *finishSemaphore, constructed with initial value 0 if joinable
Addition to Thread::Fork

if (joinable)
   kernel->currentThread->joinableList->Append(this)

Note. Put this thread to the joinable list of the current thread (the thread calling Fork)
Thread::Join

1. ASSERT(joinable);
2. joinSemaphore->P();
3. kernel->currentThread->joinableList->Remove(this);
4. finishSemaphore->V();
Addition to Thread::Finish

1. if (joinable)
   
   joinSemaphore->V()
   finishSemaphore->P()

2. while (! joinableList->isEmpty())
   
   Thread *t = joinableList->RemoveFront()
   t->finishSemaphore->V()
An implementation using lock and semaphores.

Add a new class: (see PendingInterrupt, Condition)

PendingAlarm

- when, wake-up time
- alarmSemaphore, initial value 0

Add a sorted list for pending alarms

```
SortedList<PendingAlarm> *pendingAlarmList;
```

Implement a PendingCompare function required by SortedList.
Modify Alarm constructor.

1. Construct timer
   //********************** change **********************
2. Construct a sorted list PendingAlarmList, passing PendingCompare function
3. Construct a lock for the list
   //********************** end of change **********************
Alarm::WaitUntil (see Condition::Wait and Interrupt::Schedule)

1. Calculate wake-up time, when
2. Construct a pendingAlarm passing when, which constructs a semaphore
3. Acquire the lock for the pending alarm list
4. Insert the pendingAlarm to the list
5. Release the lock
6. SetOn the alarm by calling alarmSemaphore→P() (We could have a public function PendingAlarm::SetOn to call P() and keep the semaphore private)
7. Delete the pendingAlarm

By using semaphore, we don’t directly call Sleep(), which requires interrupts off.
WaitUntil

Modify Alarm::CallBack (see Interrupt::CheckIfDue)

1. Get interrupt
2. Get status
   //************* change ***************
3. while pendingAlarmList not empty and its front alarm is due
   Remove the front pending alarm from the list
   SetOff alarm (call semaphore−>V())
   //*************** end of change **************
4. if status is not idleMode
   YieldOnReturn