CS162 Discussion #2: The Fun Begins!

About Your TA

Name:	Angela Juang	
Email:	juang.angela+cs162@gmail.com	or
	cs162-ta@imail.eecs.berkeley.edu	
Website:	http://inst.eecs.berkeley.edu/~cs162-ta	
Sections:	Friday	10-11am, 6 Evans
		11-12pm, 2 Evans
Office Hours:	Tuesday	3-4pm, 651 Soda
	Thursday	2-3pm, 651 Soda

Announcements

- Make sure you join the newsgroup! (ucb.class.cs162)
- Get a reader at Copy Central (on Hearst) as soon as possible
- Project 1 starts next Tuesday, 09/14
    - Project 1 initial design is due Monday, 09/27 – start early!

Keeping Things in Context

"Context switch" is a general term to describe the process of saving and restoring CPU state so that streams of execution can be resumed later in time. There are two main types of context switches: hardware context switches and software context switches. Hardware context switches, also called interrupts, are caused by a signal from hardware to the kernel indicating that some event (keystroke, mouse, timer, etc.) has occurred. When a hardware interrupt happens, execution goes directly to the interrupt handler. In contrast, a software context switch occurs when a thread yields execution to another thread. In this case, the kernel is responsible for choosing the next thread to run.

As we covered briefly last week, each thread has its own execution stack; a context switch from one thread to another will not affect the previous thread's stack outside of taking the necessary steps to yield. This allows threads to resume execution where they left off the next time they get CPU time. However, context switches may happen at any time during a thread's execution. What happens if the thread is in the middle of accessing some resource when it is forced to yield to another thread that also wants to access the same resource?

For example, consider a simple case where two threads each want to: 1) check how many items are in a list, and 2) if there are still items in the list, remove one. Now suppose the list has one item and Thread A checks, sees that there is one item, and is about to remove that item when a context switch gives execution over to Thread B. Thread B does the same, goes ahead and removes the item, and then yields back to A. Thread A thinks it has already passed the check and resumes trying to remove an item from the list…but there's none left! We can see that if

more than one thread is trying to access the same resource, an unlucky choice in scheduling could cause a lot of problems. Therefore, some kind of synchronization is needed to ensure that context switches won't cause threads to jeopardize each other. We will be covering synchronization in more detail in the coming weeks.

Subversion is a Good Thing

Since you will be working with a moderately large group for the rest of the semester, you will probably find that it gets a little harder to coordinate your modifications to project code. This is where Subversion (SVN) comes in. SVN is the version control system (similar to using git or CVS, if you've used those before) your group will be using for the projects. With SVN, your entire group will have its own repository in which your code is stored, and each member can check out and commit code to the repository individually. This system will help you:

- Revert to an earlier version of your code if something goes wrong
- Merge code changes easily, and resolve conflicts in code edits
- Know who contributed what code
- Let group members work on code on their own schedule

Here are some useful resources:

- *SVN Complete Reference* (http://svnbook.red-bean.com/en/1.1/ch09.html)
- *Subclipse Eclipse Plugin* (http://subclipse.tigris.org/servlets/ProjectProcess?pageID=p4wYuA)
- *Tortoise SVN Windows Client* (http://tortoisesvn.net/)

Your group's repository, complete with the Nachos source, is located at:

`https://isvn.eecs.berkeley.edu/cs162/groupXX/trunk/nachos`

where "`XX`" should be replaced with your group number.

Make sure you meet with your group early and often, and keep communication open so each member knows what's going on and what he/she should be doing. Keep in mind that it's not necessary for each member of your group to contribute in the same way – play to each member's strengths! (No, this does not mean that one of your members can say that he has no strengths and just slack off.)

If you do begin to have concerns with any part of your group, please talk to Angela about it as soon as possible before it becomes a problem. You are welcome to arrange for either private or group meetings at any point in the semester to talk about concerns with group organization or projects – check Angela's website for details on how to request a meeting!