CS533 Concepts of Operating Systems

Class 5

Integrated Task and Stack Management
Questions

- What do these terms mean?
  - process, thread, kernel-level thread, user-level thread, fiber?
  - user-level thread stack, user stack, kernel stack?
  - Address space?
Questions

- In what way are user-level threads “faster” than kernel-level threads?
  - Aren’t we just layering abstractions on abstractions?

- Why can’t you do preemptive user-level thread scheduling without special kernel support?
  - Does this mean you can only do sequential task management?
  - How would you support cooperative task management?
Questions

- Is synchronization needed for cooperative user-level scheduling on uni-processors?
  - What precautions must you take?
  - Is synchronization needed on multiprocessors?

- How are synchronization operations such as spin-locks and semaphores implemented, i.e., how do you spin and how do you block?
  - For user level threads?
  - For kernel level threads?
Questions

- How might a user-level thread library support synchronous I/O calls for its threads without losing control of the CPU due to blocking?
  - How does this approach behave during page-faults?
  - Can it deal with preemptions?

- Do user-level thread libraries do “stack ripping”? 

CS533 - Concepts of Operating Systems
Questions

- What stack management model do most OS kernels export?
  - How are I/O and page faults events handled?

- What differences are there between virtual CPUs (threads) and real CPUs (hardware)?
  - How are I/O and page fault events handled?
  - How is time managed?