Problem Set 1

Due Sunday January 15, 11:59pm

1. Identify the first line of xv6 code that is executed in the kernel when a system call occurs, when an interrupt occurs, and when an exception occurs.

2. A system call, such as UNIX open, ultimately leads to a trap into the operating system kernel. Find where in xv6 the system call is invoked.

3. Why can't we use the native C compiler libraries to build user programs to run on xv6? Likewise, why can't we use those libraries in xv6 kernel mode?

4. xv6 provides a C library printf function for use by the xv6 applications, and a separate cprintf function for use by the kernel. Why?

5. Where is the first line of code for constructing an xv6 trapframe? How large is an xv6 trapframe? Why?

6. In xv6, when a user program (such as the shell) returns from main, what is done with the value it returns?

7. Do xv6 chapter 1, problem 1.

8. Add a tracing utility to xv6 to print (to the console) every system call as it occurs and its return value.

9. Add an upcall mechanism to xv6 to call up to user space. Add a system call, alarm(procptr, interval), that sets up a periodic upcall to procptr every interval time ticks, in other words, the user-level equivalent of a hardware timer.