

Quiz 7

CSC 256/456

December 13, 2011

1. **Privileged instructions:** Which of the following instructions should be privileged (and only allowed to execute in kernel mode)?
 - Load a value from a memory address to a general-purpose register.
 - Load a value from a memory address to an entry in the translation look-aside buffer (TLB).
 - Flush/clear the content in the translation look-aside buffer (TLB).
 - Read the current value in the program counter (PC) register.
 - Set a new value in the program counter (PC) register.
 - Turn off interrupts.
 - Call a subroutine — pushes the return address onto the stack and jumps to the subroutine.
 - Return from a subroutine — pops the return address off the stack and jumps to it.
 - x86 *MMX* (Matrix Math eXtensions) instructions — access and perform arithmetic operations on a special set of 64-bit registers to speed up multi-media data processing.
 - Halt the CPU.

2. The following algorithm, developed by Dekker, is the first known correct software solution to the critical section problem for two processes. The two processes, P_0 and P_1 , share the following variables:

```
bool flag[2]; /* initially both flags are false */
int turn = 0 or 1;
```

The following program is for process P_i ($i=0$ or 1), with P_j ($j=1$ or 0) being the other process:

```
for (;;) {
    flag[i] = true;
    while (flag[j])
        if (turn==j) {
            flag[i] = false;
            while (turn==j); /* spin wait */
            flag[i] = true;
        }
    ...
    critical section
```

```
    ...  
    turn = j;  
    flag[i] = false;  
    ...  
    remainder section  
    ...  
}
```

Explain that the algorithm satisfies all three requirements (mutual exclusion, progress, bounded waiting) for the critical section problem.

3. **Blocking and deadlock.**

(a) Programs may request some free memory during the course of execution. Without special care, a request for free memory may block for a potentially long period of time (several milliseconds). Explain how this can happen.

(b) Blocking inside an interrupt handler is a recipe for deadlock. Explain how deadlock may occur when an interrupt handler makes a potentially blocking request for free memory.

4. **Reliability.** Device drivers are probably the buggiest part of an OS kernel since their developers (hired by device manufacturers) may not be as diligent as the core kernel developers.

- (a) In order to prevent buggy drivers from overwriting other parts of the kernel, we let each device driver have its own kernel-level page table. A driver's page table makes clear that it can only write to a designated set of pages. When the CPU control transfers in and out of a driver, the kernel switches page tables appropriately. Explain that this scheme can only protect against buggy drivers, not malicious drivers.

- (b) In addition to overwriting other parts of the kernel, describe another potential harmful consequence of buggy (not malicious) drivers on the rest of the operating system.

- (c) The part of device drivers that uses privileged I/O instructions or accesses privileged device registers must run in the kernel mode, but other parts may run in the user mode. Running parts of device drivers in user mode may improve the OS reliability. Describe any performance problem with this scheme.