CS503 Spring 2019 - Quiz 4B (10 pts)

Answer the questions in the spaces provided. If you run out of room, continue on the back.
Provide clear answers and justify statements where required.
Unless otherwise noted, assume that questions refer to C code, AT&T assembly syntax and the x86 architecture.

1. (2 points) Name:

2. (6 points) State whether the following statements are true of false. **Explain the false statements.**

   (a) (1 point) Paging divides the logical address space into blocks called pages.
   - ◯ True
   - ◯ False. Explanation:

   Solution: True.

   (b) (1 point) The "RDTSC" instruction returns the number of seconds elapsed since year 2000.
   - ◯ True
   - ◯ False. Explanation:

   Solution: False. Returns the number of cycles. Optional: CPU cycles since last reset.

   (c) (1 point) In Xinu, the delta list stores in the key field the absolute time of the scheduled events.
   - ◯ True
   - ◯ False. Explanation:

   Solution: False. The key stores the deltas between scheduled events/current time.

   (d) (1 point) File systems generally operate on two disk partitions, one of which stores the superblock.
   - ◯ True
   - ◯ False. Explanation:

   Solution: False. Operate on one partition.

   (e) (1 point) Traditional Unix file systems rely on a single hierarchical namespace.
   - ◯ True
   - ◯ False. Explanation:

   Solution: True.

   (f) (1 point) Interrupt handlers use the "RET" instruction to return.
   - ◯ True
   - ◯ False. Explanation:

Solution: False. Use IRET.

3. (2 points) Message passing and shared memory are two different IPC approaches with different advantages and disadvantages for application developers. List 2 advantages of using message passing and 2 other advantages of using shared memory.

Solution: Message passing: explicit control over communication; no need to synchronize explicitly accesses. Shared memory: low access overhead, ability to operate on pointers directly / ease of sharing complex data structures

4. (2 points (bonus)) Some operating system, like Linux, support the concept of signals. Signals allow the operating system to send events (e.g., user typed Ctrl+C) to applications, similarly to interrupts, which allow hardware to send events to the operating system. In such OSs, the application can register a function, the signal handler, that is asynchronously executed in the context of one of the application threads (i.e., it does not spawn a new thread) and in the address space of the application. Explain and justify whether the application developer needs to worry about synchronization when writing signal handlers if there is only one application thread. In such situation can signal handlers cause deadlocks or other synchronization problems? Explain.

Solution: Developers do not need, and should not use, synchronization functions such as locks and semaphores inside signal handlers when there is only one thread: blocking or busy waiting inside a signal handler would cause the application to hang because there is only one thread. However, developers should ensure that if signal handlers share data with the rest of the code some precautions are taken. In particular, shared data should only be atomically modified by the rest of the code to ensure it is consistent anytime a signal handler can be invoked.