CS 354 (Park) **Final** **May 2 (Tue.), 2023**

Remarks: Keep the answers compact, yet precise and to-the-point. Long-winded answers that do not address the key points are of limited value. Binary answers that give little indication of understanding are no good either. Time is not meant to be plentiful. Make sure not to get bogged down on a single problem.

PROBLEM 1 (40 pts)

(a) Describe a simple example of a deadlock for an app comprised of two processes. How would a kernel detect that the two processes are deadlocked? What is the rationale for a kernel not to provide deadlock detection service to app processes? Besides careful programming, what technique can a programmer who codes the app use to guarantee that the two processes will not deadlock?

(b) What are the two main benefits that DMA provides for supporting video streaming from a digital video camera (e.g., webcam) by modern kernels? What are the two items commanded by the lower half to a DMA controller to instruct it what to do? Suppose a producer/consumer kernel buffer is shared between lower half and upper half for video streaming. Who is the producer (or writer) of the kernel buffer? Who is the consumer (or reader)? Be precise in characterizing the "who" parts based on our discussion of how IPC coordination/synchronization mechanisms are borrowed for device I/O.

PROBLEM 2 (40 pts)

(a) What are the similarities and differences between FAT and XINU file systems? What is the main weakness of both file systems? Despite the weakness, why are FAT file systems relevant in today's real-world environment? Why is FAT not suited as the main file system of commodity kernels? How do UNIX/traditional file systems address the weakness? What are the resultant concrete benefits?

(b) What is the meaning of tickful kernel? Why is the 1 msec granularity counter implemented in lab assignments in XINU potentially inaccurate as a wall clock timer due to how XINU is designed? How would XINU's upper and lower halves need to be changed to mitigate this inaccuracy? What is the meaning of a tickless kernel? Describe the conditions under which a tickless design may be preferable to a tickful design, and why. What is the role of an interval timer in a tickless kernel?

PROBLEM 3 (20 pts)

What is the technical meaning of page fault? What is the rationale behind kernels handling page faults instead of hardware? Assuming a free frame is available, what tasks are performed by a kernel when a page fault occurs up until the moment the page faulting process becomes ready to resume executing? How is context-switching overhead affected in kernels that support virtual memory? Why is the practice of mapping kernel memory to the same virtual memory address space across all processes beneficial for reducing context-switch overhead?

BONUS PROBLEM (10 pts)

When discussing design and implementation of kernels we considered correctness and performance as key criteria where the latter pertained to overhead (i.e., time complexity) of algorithms to carry out a task. Describe two examples we discussed where "performance" had a different aspect or meaning, not related to overhead of code implementing an algorithm.