1. Which of the following is true of a device driver?
   (b) It is a part of the systems software

2. Which of the following set of instructions is likely to be present at the kernel mode ISA, but not at the user mode ISA?
   (a) IN and OUT (read from or write to an I/O interface module register) CALL and RETURN (call or return from a subroutine)

3. A DMA (direct memory access) operation is
   (f) initiated by the device driver, but carried out by the DMA controller

4. Which of the following functions/modules is unlikely to be entered when the user-level program calls the function `scanf()`?
   (d) process control system

5. Which of the following functions/modules is unlikely to be used to read from the keyboard?
   (c) DMA

6. Which of the following is the most likely reason for disabling interrupts when entering the system call interface routine?
   (c) Some book-keeping work (saving registers, etc) is required to be completed before the system can accept another interrupt.

7. A DMA controller has a 32-bit status register and three 32-bit data registers. The status register is assigned to consecutive memory locations starting at label `DMA_status`. The three data registers are assigned to consecutive memory locations starting at label `DMA_data`. The controller interface works as follows. The controller is ready to accept a new command, only if the most significant bit (MSB) of its status register has been set to 1. When a new command is to be given to the DMA controller, the DMA initiation device driver writes the starting memory address in the first data register, the sector number in the second data register, and the number of bytes to be transferred in the third data register. After updating the data registers, the device driver sets the MSB of the status register to 0, and the next significant bit to 0 or 1 (0 for reading from IO device, and 1 for writing to IO device).

Write a MIPS-I assembly language device driver that can be used to initiate a DMA operation. This device driver is called with 4 parameters: $a0 contains the operation (0 for read; 1 for write); $a1 contains the starting address in memory; $a2 contains the sector number in the disk; and $a3 contains the number of bytes to be transferred. Assume that the return address is present in $ra.
.ktext

busy_wait:    # Check if DMA controller is ready to accept a new command
lw   $t0, DMA_status
li   $t1, 0x80000000
and  $t2, $t0, $t1
beq  $t2, $zero, busy_wait

# At this point DMA controller is ready
# Update DMA controller’s data registers
la   $t2, DMA_data
sw   $a1, 0($t2)
sw   $a2, 4($t2)
sw   $a3, 8($t2)

# Update DMA controller’s status register
# Clear MSB
# Set next significant bit to 0 or 1 indicate read/write
# This information is available in $a0
li   $t3, 30
sll  $t3, $a0, $t3
li   $t4, 0x3fffffff
and  $t0, $t0, $t4
or   $t0, $t0, $t3

# Return
jr   $ra