1) (5pts) Why is polling or “spin-waiting” a bad idea for input/output programming?

2) (5pts) What would happen on a computer system that used a spin-wait loop to print a character if a printer jammed?

3) (5pts) Does an assembly language programmer have any control over what type of I/O implementations are offered?

4) (5pts) How is a trap different from an interrupt?

5) (5pts) What is an exception handler?

6) (5pts) Why should the exception handler be privileged code?
7) (10pts) Assign priorities to the following exception conditions from most important to least important and justify the ordering. Explain your reasoning.
   a) syscall
   b) clock
   c) arithmetic overflow

8) (5pts) What would happen if the exception handler returned control to a user program without re-enabling interrupts?

9) (5pts) If a re-entrant exception handler did not disable interrupts as it prepared to return control to the interrupted program, and was in the process of restoring the registers to their previous values when another interrupt occurred, could the registers still be restored correctly after the second interrupt was handled? What could go wrong?

10) (5 pts) What are the advantages and disadvantages of implementing polling? How about Interrupts?

11) (5 pts) What is the point of a pipelined architecture? Why do it?

12) (5 pts) Why is it harder to pipeline a CISC architecture like the HC11 and x86 verses there RISC counterparts like the MIPS and PowerPC?
13) (10 pts) Where are the dependencies in these instructions in the DLX pipeline? How can we fix the problems?

    add $4, $5, $6
    subi $8, $4, 4
    lw $11, ($8)