CS 2200 Homework 4

Spring 2019

Rules:

- Please print a copy of the assignment and hand write your answers. No electronic submissions are allowed. **Please print as one double-sided page. Do NOT staple multiple sheets together.**
  There will be a **40** point penalty if you do not.
- This is an individual assignment. You may discuss concepts but not the answers.
- Due Date: **13th February 2018 – 6:00 PM**. Bring your BuzzCard. Show up on time.

Name:_____ **GT Username:**_____ **Section:**_____

1. Explain why it wouldn't be a good idea to disable interrupts at the beginning of the interrupt handler and not enable them until RETI was called at the end?

2. What two actions does the RETI instruction perform? Why must these two actions be done in a single instruction, as opposed to allowing the programmer to use multiple instructions?

3.

Event	Synchronous/Asynchronous (Circle One)		Exception, Trap, or Interrupt (Circle One)		
Divide by Zero	S	A	E	T	I
Mouse Click	S	A	E	T	I
Opening a .pdf File	S	A	E	T	I
Segfault from accessing a null pointer	S	A	E	T	I
Keyboard Key Press	S	A	E	T	I

4. Complete the following interrupt handler by filling in the missing assembly code. Assume the processor has already placed PC in $k0, disabled interrupts, and set the PC ot the first instruction in int_handler.

   We've given you the main logic of the handler, which simply increments a counter in memory. **Your completed handler must allow nested interrupts, i.e., another interrupt should be permitted to come in while the handler is running.** Do not leave interrupts disabled longer than absolutely necessary. Only save the minimum of registers absolutely necessary.

   Here are the new instructions we added to LC-2200 for interrupt handling:
   - **EI** - enable the ability for interrupts to occur
   - **DI** - disable the ability for interrupts to occur
   - **RETI** - set PC = $k0 and enable interrupts

```
int_handler:
addi   $sp, $sp, _____
sw     _____, 0x0($sp)
_____

addi   _____, _____, -2
sw     _____, 0x1($sp)
sw     _____, 0x0($sp)

! main interrupt handler logic here

_____ $t0, 0x1($sp)
_____ $t1, 0x0($sp)
addi   _____, _____, 2

_____
lw       $k0, _____(____)
addi   _____, _____, 1
_____
```