Name: **Id#**

COE 301/ICS 233, Term 172

Computer Architecture & Assembly Language

Quiz# 2

Date: Thursday, Feb. 22, 2018

Q1. Fill in the blank in each of the following questions:

(1) Assuming 12-bit unsigned number representation, the binary number 1111 1111 0000 is equal to the decimal number _____.

(2) Assuming 16-bit signed 2`s complement representation, the hexadecimal number FEA0 is equal to the decimal number _____.

(3) The pseudo instruction *li $t0, 0x12345678* is implemented by the following minimum MIPS instructions:

_____
_____
_____

(4) The pseudo instruction *neg $s2, $s1* ($s2 is computed as the negative value of $s1) is implemented by the following minimum MIPS instructions:

_____
_____
_____

(5) The pseudo instruction *rol $s0, $s0, 4* ($s0 is rotated to the left by 4 bits and stored in $s0) is implemented by the following minimum MIPS instructions:

_____
_____
_____

(6) Assuming that $a0 contains an Alphabetic character, the instruction _____ will make the character stored in $a0 always upper case. Note that the ASCII code of character 'A' is 0x41 while that of character 'a' is 0x61.

(7) Assuming the following data segment, and assuming that the first variable X is given the address **0x10010000**, then the addresses for variables Y and Z will be _____ and _____.

```
.data
X:    .byte  10, 11, 12, 13, 14
Y:    .half  15, 16, 17, 18
Z:    .word 19, 20
```

(8) To multiply the **signed** content of register $t0 by 112 without using multiplication instructions, we use the following minimum MIPS instructions (HINT: 112=16*7):

_____
_____
_____
_____

(9) To allocate 10 words, each initialized by 0, we use the following assembler directive _____.

(10) The MIPS system call for printing a string given below prints the following:

_____
_____
_____

Note that the ASCII code for the line feed character is 10 and the ASCII code for the carriage return character is 13.

```
MSG: .ascii  "Quiz#2 "
     .byte 10
     .ascii "COE 301/ICS 233 "
     .asciiz "is easy !! "

      li $v0, 4
      la $a0, MSG
      syscall
```