July 22, 2007

COMPUTER ENGINEERING DEPARTMENT

ICS 233

COMPUTER ARCHITECTURE & ASSEMBLY LANGUAGE

Major Exam I

Summer Semester (063)

Time: 7:00-9:30 PM

Student Name : ____

Student ID.   : ____

Question	**Max Points**	**Score**
Q1	**40**	
Q2	**15**	
Q3	**18**	
Q4	**12**	
Q5	**15**	
Total	**100**	

Dr. Aiman El-Maleh

[40 Points]

(Q1) Indicate whether the following is **true** or **false**, and if it is false **correct** it (correct the answer and not the question):

(1) (True, False) The smallest (negative) number that can be represented using 8-bit 2`s complement in hexadecimal is FF and the largest positive number in hexadecimal is 7F.

(2) (True, False) Assume that the CPU has just read a 32-bit instruction from the address 0x00400000. Then, the address of the next instruction that this CPU is going to read is 0x00400001.

(3) (True, False) Assuming 8-bit representation of numbers, the binary number 10100100 is equal to -36 in sign-magnitude representation, -91 in 1`s complement representation, and -92 in 2`s complement representation.

(4) (True, False) The following assembler directive allocates 1 word initialized by 10.

```
X: .word 1:10
```

(5) (True, False) With a 32-bit address bus and 32-bit data bus, the maximum memory size than can be accessed by a processor is 4MByte and the maximum number of bytes that can be read or written in a single cycle is 8 Bytes.

(6) (True, False) Assuming variable Array is defined as shown below:

```
Array: .word 0x00000010, 0x00000020
```

The content of register \$t0 after executing the following sequence of instructions is 0x00000020.

```
la $t0, Array
lw $t0, 1($t0)
```

(7) (True, False) The instruction set architecture of a processor consists of its control unit, data path, memory, and the instruction set.

(8) (True, False) Given a magnetic disk with the following properties:

- Rotation speed = 7200 RPM (rotations per minute)
- Average seek = 8 ms, Sector = 512 bytes, Track = 200 sectors

The average time to access a block of 64 consecutive sectors is 13.5 ms.

(9) (True, False) Assuming the following data segment, and assuming that the first variable X is given the address **0x10010000**, then the address for variable Y will be **0x10010005.**

```
.data
X:      .byte  10, 11, 12, 13, 14
Y:      .word  15
```

(10) (True, False) The code given below prints the statement: *Exam1*

```
MSG: .ascii  "Exam1"
     .asciiz " ICS 233"

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

(11) (True, False) Assume that the instruction j NEXT is at address 0x00400020 in the text segment, and the label NEXT is at address 0x00400010. Then, the address stored in the assembled instruction for the label NEXT is 0x0400010.

(12) (True, False) Assume that the instruction beq $t0, $t1, NEXT is at address 0x00400020 in the text segment, and the label NEXT is at address 0x00400010. Then, the address stored in the assembled instruction for the label NEXT is 0xfffb.

(13) (True, False) After executing the instruction sll $t0, $t0, 2, the content of register $t0 is equal to 2*$t0, for both signed and unsigned content.

(14) (True, False) The code given below implements the conditional statement

if (($t0 < 1) AND ($t1 > 100)) Then $t2=0.

```
        slti $t3, $t0, 1
        bne $t3, $zero,  Zero_index
        li $t3, 100
        slt $t3, $t3, $t1
        beq $t3, $zero, End_if
Zero_index:
        xor $t2, $t2, $t2
End_if:
```

(15) (True, False) Assuming that $a0 contains an Alphabetic character, the instruction *andi $a0, $a0, 0xdf* will guarantee that the character in $a0 is an upper case character. Note that the ASCII code of character ‘A’ is 0x41 while that of character ‘a’ is 0x61.

(16) (True, False) Assume you are in a company that will market a certain IC chip. The cost per wafer is $4000, and each wafer can be diced into 2000 dies. The die yield is 60%. Then the cost per good die is $2.

(17) (True, False) Assume that \$t0=0xffff8111 and \$t1=0xfffff265. Executing the instruction *subu \$t0, \$t0, \$t1* produces correct result in \$t0 assuming both signed and unsigned number representation.

(18) (True, False) The difference between *add* and *addu* instructions is that *add* should be used for addition of signed numbers while *addu* should be used for addition of unsigned numbers.

(19) (True, False) Executing the following sequence of instructions produces the value 0x0000009b in \$s2.

```
li      $s1, 5
sll     $s2, $s1, 5
subu    $s2, $s2, $s1
```

(20) (True, False) Assuming that \$s1=0xabcd1234 and \$s2=0xffff0000, executing the instruction *nor \$s0,\$s1,\$s2* produces the value 0xffff1234 in \$s0.

[15 Points]

(Q2) Using only basic MIPS instructions, write the shortest sequence of instructions to implement each of the following pseudo instructions:

1. *li $t0, 0x12345678* #$t0 is loaded with the immediate value 0x12345678

2. *not $t0, $t1* #$t0 is loaded with the 1's complement value of $t1

3. *bge $t0, $t1, Next* # branch to Next if $t0 is greater than or equal $t1

4. *abs $t0, $t1* #$t0 is loaded with the absolute value of $t1

5. *rol $t0, $t0, 5* #$t0 is rotated to the left by 5 bits and stored in $t0

[18 Points]

(Q3) Answer the following questions. Show how you obtained your answer:

(i) Given that **TABLE** is defined as: **TABLE: .ascii "Ahmad Ali Anas"**

Determine the content of register **$t0** after executing the following code:

```
        xor $t0, $t0, $t0
        li $t1, 14
        la $t2, TABLE
        addi $t2, $t2, -1
Next:   beq $t1, $zero, ENL
        addi $t2, $t2, 1
        lbu $t3, ($t2)
        ori $t3, $t3, 0x20
        li $t4, 'a'
        addi $t1, $t1, -1
        bne $t3, $t4, Next
        addi $t0, $t0, 1
        j Next
ENL:
```

(ii) Given that **TABLE** is defined as shown below**:**

TABLE: .space 33

Determine the output produced after executing the following code:

```
        li $t0, 0xabcde765
        li $t1, 32
        la $t2, TABLE
AGAIN:  li $t3, '0'
        rol $t0, $t0, 1
        andi $t4, $t0, 1
        add  $t3, $t3, $t4
        sb $t3, ($t2)
        addi $t2, $t2, 1
        addi $t1, $t1, -1
        bne $t1, $zero, AGAIN
        la $a0, TABLE
        li $v0, 4
        syscall
```

(iii) Given that TABLE is defined as shown below, determine the content of TABLE after executing the following code:

TABLE: .word 1, 2, 3, 4, 5, 6, 7, 8

```
        la $t0,  TABLE
        addi $t1, $t0, 28
        li $s0, 4
Again:
        lw $t2, ($t0)
        lw $t3, ($t1)
        sw $t2, ($t1)
        sw $t3, ($t0)
        addi $t0, $t0, 4
        addi $t1, $t1, -4
        addi $s0, $s0, -1
        bne $s0, $zero, Again
```

[12 Points]

(Q4) Write a MIPS assembly program to do the following using the smallest possible number of instructions. Ask the user to enter two integers and then display their sum according to the format given below.

Sample Execution:

```
Enter an integer: -2
Enter another integer: 20
The sum of -2 and 20 = 18
```

[
15 Points]

(Q5) Write a MIPS assembly program to sort an array of integers (i.e. 32-bit signed numbers) in an **ascending** order using **BubbleSort** algorithm. Minimize the number of instructions used.

The pseudocode for the **BublleSort** algorithm is given below:

```
BubbleSort (ArrayPointer, ArraySize)
    Status = Unsorted
    #comprisons = ArraySize-1
    while (#comparisons<>0 AND status = Unsorted)
        Status = Sorted
        for (i= 0 to #comparisons)
            if (Array[i] > Array[i+1])
                swap ith and (i+1)th elements of the array
                Status = Unsorted
            end if
        end for
    #comparisons = #comparisons – 1
    end while
end BubbleSort
```

Clearly indicate the registers used for each variable. Store the array to be sorted in variable Array as defined below.

Array: .word 10, 2, 0, 15, 25, 30, 7, 22