Assembly Programming (CSE3030-01), Midterm Exam, Spring 2018

18:00-19:15pm, April 25 2018 (Thursday), Instructor: Dr. Youngjae Kim

Student ID:_____ Name:_____

1. Basics of Processor Architecture (25 points)

1) [2pt] Explain terms of MSB and LSB.
2) [2pt] What is the sum of each pair of binary integers?
   a. 01110101+10101100
   b. 11010101+01101011
3) [2pt] How many bytes are in each of the following data types? Suppose word size is 2 bytes.
   a. Doubleword
   b. Quadword
4) [3pt] Convert hexadecimal number F1ABC1 to octal and binary.
5) [4pt] What is the decimal representation of the following signed binary numbers?
   a. 11110000
   b. 11001100
6) [4pt] What is the 8-bit binary (two's complement) representation of each of the following signed decimal integers?
   a. -25
   b. -75
7) [3pt] What are the five basic steps in the instruction execution cycle?
8) [3pt] Suppose a five-stage pipelined processor. Explain five-state pipeline to execute an instruction, and how the processor gains performance by pipelining.
9) [2pt] Define multitasking in OS.

2. Assembly Language Grammar (10 points)

Answer the following questions for the program shown below.

```
TITLE Add and Subtract (AddSub.asm)
INCLUDE Irvine32.inc
.code
main PROC
mov eax,10000h    ; EAX = 10000h
add eax,40000h    ; EAX = 50000h
sub eax,20000h    ; EAX = 30000h
call DumpRegs     ; display registers
exit
main ENDP
END main
```

1) [2pt] What does the .CODE directive identify?
2) [2pt] How are the CPU registers displayed?
3) [2pt] Which statement halts the program?
4) [2pt] Which directive begins a procedure?
5) [2pt] Which directive ends a procedure?

3. Simple Assembly Language (20 points)

1) [4pt] What will be the value in the destination operand after the following lines execute?

```
mov al, 09h
add al, 0cfh
```

2) [5pt] What will be the value in EAX after the following lines execute?

```
mov eax, 1002FFFFh
neg ax
add al, 0cfh
```

3) [6pt] What will be the value of EAX and the Sign flag, Overflow flag, Carry flag, after the following lines execute?

```
mov eax, 5
sub eax, 6
```

4) [5pt] In the following code, the value in AL is intended to be a signed byte. Explain how the Overflow flag helps, or does not help you, to determine whether the final value in AL falls within a valid signed range.

```
mov al, -1
add al, 130
```

4. True/False questions (20 points)

1) [5pt] Is it possible to set the Overflow flag if you add a positive integer to a negative integer? If it is true, give an example to justify your answer. Else, explain why it can't be.
2) [5pt] Will the Overflow flag be set if you add a negative integer to a negative integer and produce a positive result? If it is true, give an example to justify your answer. Else, explain why it can't be.
3) [5pt] Is it possible for both the Sign and Zero flags to be set at the same time? Justify your answer using examples.
4) [5pt] Can't the NEG instruction set the Overflow flag? Justify your answer.

5. Understanding Assembly Program (12 points)

```
.data
ArrayB     byte   'b3EB', -124, 31h, 31, 1 DUP(-9)
ArrayW1    word   '92', 23, 05AA5h, -5
ArrayW2    word   2 DUP(-14), 31, "31"
ArSize = $ - ArrayW1
ArrayD1    dword  '9Ba0', "2048"
ArrayD2    dword  2048, ArSize
```

Suppose the following program executes with respect to data declaration above. Write the output of destination operand for every instruction execution. If an instruction is wrong, then, mark 'X'. Also, assume that the preceding instruction does not affect the following instruction. Use supplementary material for ASCII table if necessary. (Each line is 2 points.)

```
movzx ax, [ArrayB + 1]          ; ax=
movsx ax, [ArrayB + 4]          ; ax=
mov   ax, ArrayW1               ; ax=
mov   ArrayD1, ArrayD2          ; ArrayD1=
mov   eax, [ArrayD1 - 5]        ; eax=
mov   ax, [ArrayW1 + 0Fh]       ; ax=
```

6. Write in Assembly Language (13 points)

1) [2pt] Declare a string variable (named favColor) containing the name of your favorite color. Initialize it as a nullterminated string.
2) [2pt] Declare an array of 50 unsigned bytes (named aArray) and initialize all elements to zero. Use DUP operator.
3) [9pt] Write a program that uses the variables below and MOV instructions to copy the value from bigEndian to littleEndian, reversing the order of the bytes. The number's 32bit value is understood to be 12345678 hexadecimal.

```
.data
bigEndian BYTE 12h, 34h, 56h, 78h
littleEndian DWORD ?
```