ENCS336 – 1st Exam

Date: 11/11/2007

ID:____________________ **Name:**____

□ **section 1** □ **section 2**

Question 1 (20 marks) **:**

a) TRUE or FALSE :

	TRUE	FALSE	
1	○	●	*Organization* is those attributes visible to the programmer and *Architecture* is how features are implemented.
2	○	●	All Intel x86 family share the same basic organization in order to maintain code backwards compatibility
3	○	●	ENIAC was a binary machine programmed manually by switches
4	●	○	RTL is a symbolic notation used to describe the microoperation transfers among registers
5	○	●	In Random Access Memory, accessing a word requires different time dependent on the location of the word
6	●	○	The Adder is the basic module in the ALU
7	●	○	A 4 bit binary decrementer can be implemented by adding 1111 to the desired register
8	○	●	PC holds address of the current instruction to fetch
9	●	○	The number of bits required for the op-code depends on the total number of operations
10	●	○	The current Peripheral Component Interconnection standard allows the use of up to 64 data lines at speed 66MHz

b) Multiple choices:

1. The ASCII code for the character '7' is 37H. What is the ASCII code for the character '1'?

○2FH	○47H	●31H	○3EH

2. What will be the value in the DL register after executing the following two instructions?

```
MOV DL, 7EH
CMP  DL, 6
```

○77H	○85H	○7FH	●7EH

3. The register transfer statement R1 ← R1 + R2' + 1 describes

○ADD	●SUB	○INC	○DEC

4. Assume the initial value of a register is $10011010_{(2)}$. If we apply an SAR operation followed by a ROL operation to this register, then the final contents of this register will be:

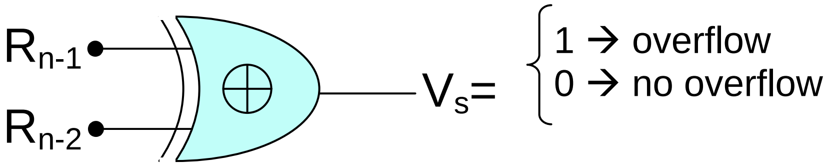
○$10011010_{(2)}$	○$11100110_{(2)}$	●$10011011_{(2)}$	○ No change

Shift **A**rithmetic **R**ight.
This section will be **Bonus**

Question 2 (25 marks) :

a) Show a simple way to detect if an arithmetic shift-left will result an overflow?

An overflow flip-flop Vs can be used to detect an arithmetic shift-left overflow
Vs = Rn-1 ⊕ Rn-2



b) How many 16-bit words of memory can the 24-bit address identify uniquely, if memory is byte-addressable?

$2^{24}/ 2^{1} = 2^{23} = $ 8K words

c) Explain the concept of a stored program computer?

A stored program computer is one in which the machine language program is stored in memory along with data, and the computer is able to manipulate both program and data in identical ways.

d) List and briefly define two approaches to dealing with multiple interrupts?

- **Disabling interrupts while an interrupt is executing.**
- **Setting up priorities for interrupts.**

e) What is the benefit of using a multiple-bus architecture compared to a single–bus architecture?

With multiple buses, there are fewer devices per bus. This reduces <u>propagation delay</u>, because each bus can be shorter, and reduces bottleneck effects.

Question 3 (25 marks) **:**

a) Show the IEEE 754 binary representation of the number -0.75 in single and double precision?

-0.75 ➔ -0.11 binary ➔ -0.11 * 2^0 = -1.1 * 2^{-1}
Single bias ➔ 127 ➔ biased exponent is -1 + 127 = 126

-0.75 ➔ 1 01111110 100000..0

Double bias ➔ 1023 ➔ biased exponent is -1 + 1023 = 1022

-0.75 ➔ 1 01111111110 100000..0

b) What decimal number is represented by the following single precision float?

0 10000001 010000000000 ··0

Sign bit = 0
Exponent = 129
Fractions = ¼ = 0.25

+1.25 * $2^{(129-127)}$ = +5

c) Show the steps of multiply **-11** by **14** using Booth's algorithm?

A	Q	Q-1	M	Notes
00000	0111**0**	**0**	-11 = 10101	Initial value
00000	0011**1**	**0**		Shift ----------- step 1
01011	00111	0		Sub ----------- step 2
00101	1001**1**	**1**		Shift
00010	1100**1**	**1**		Shift ----------- step 3
00001	0110**0**	**1**		Shift ----------- step 4
10110	01100	1		Add ----------- step 5
11011	**00110**	0		Shift

d) In Booth's algorithm, which is better to multiply (**-11 * 14**) or (**14 * -11**)? Explain why?

10101 (-11) * 01110 (14) ➔ M * $(2^4 – 2^1)$ ➔ one addition and one subtraction only

01110 (14) * 10101 (-11) ➔ M * $(-(2^5 – 2^4) + (2^3 – 2^2) + (2^1 – 2^0))$ ➔ 3 additions and 3 subtraction

So the first one (-11 * 14) is more efficient; takes les operations.

Question 4 (30 marks) **:**

a) Show the absolute addresses formed by the following (in hexadecimal):

- SS contains 2AB4h and SP contains 24h : **2AB64h**
- CS contains 2BC3h and IP contains 3Ah : **2BC6Ah**

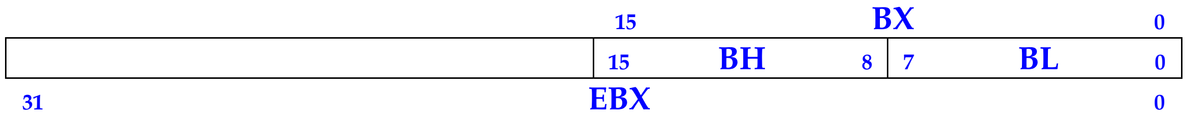
b) Assume that you don't have the instruction (**XCHG X, Y**), write an equivalent instructions using **Only** the STACK instructions?

PUSH X
PUSH Y
POP X
POP Y

c) Explain which registers used for the following purpose:

- Counting for looping : **CX**
- Multiplication and division : **DX , AX**
- Addressing segments : **CS, DS, SS, ES**
- Indication of a zero result : **ZF (Zero flag)**
- Offset address of an instruction that is to execute : **IP**

d) Show the EBX register and the size and positions of BH, BL, and BX within it?



e) Show how the value 0AC73135Fh is stored in memory on the 80x86 Processor starting at memory location 100h?

:	**:**
100	**5F**
101	**13**
102	**73**
103	**AC**
:	**:**

f) Write 8086 assembly language code to implement the following high-level construct. Assume all comparisons are made on signed numbers.

```
          :
          CMP AX, 3
          JL THEN
          CMP BX, 2
          JL THEN
ELSE:
          SUB CX, CX ;    or MOV CX, 0
          JMP ENDFIF
THEN:
          MOV CX, 1
ENDIF:
          :
```

```
IF ((AX < 3) OR (BX < 2)) THEN
   CX = 1
ELSE
   CX = 0
ENDIF
```

g) Write an 80x86 Assembly Language code which

- Prompts the user to enter two characters as input
- Prints out whether the characters are equal or not.

```
 ORG 100h
JMP START

 PROMPT  DB "ENTER TWO CHARACTERS: $"
 MSG1       DB "TWO CHARACTERS ARE EQUAL. $"
 MSG2       DB "TWO CHARACTERS ARE NOT EQUAL. $"

START:

          MOV DX, OFFSET PROMPT
          MOV AH, 09h
          INT 21h                     ; prompt the user to enter 2 chars

          ; read two chars
          MOV AH, 06h
          MOV DL, FFh
          INT 21h
          MOV  BL, AL                 ;    1st char in BL
          INT 21h                     ;    2nd char in AL
          CMP AL, BL
          JE EQUALE
NOTEQUAL:
          MOV DX, OFFSET MSG2
          MOV AH, 09h
          INT 21h                     ;  print out that the 2 chars are not equal
          JMP AFTER:
EQUAL:
          MOV DX, OFFSET MSG1
          MOV AH, 09h
          INT 21h                      ;  print out that the 2 chars are equal
AFTER:
```