# Mid I Revision

- What is the maximum memory 8088 can access?

- What is the difference between little endian and big endian formats? Which format is used by the Intel 8088 microprocessor?

- What is the size of the accumulator of a 64bit processor?

- What is the difference between an instruction mnemonic and its opcode?

- What is a label and how does the assembler differentiate between code labels and data labels?

Give the value of the zero flag, carry flag, sign flag, and the overflow flag after each of the following instructions if AX is initialized with 0x1254 and BX is initialized with 0x0FFF.

a. add ax, 0xEDAB

b. add ax, bx

c. add bx, 0xF001

- What are the contents of memory locations 200, 201, 202, and 203 if the word 0x1234 is stored at address 200 and the word 0x5678 is stored at address 202?

- Why a segment cannot start from the physical address 55555.

Calculate the physical memory address generated by the following segment offset pairs.

a. 1DDD:0436

b. 1234:7920

c. 74F0:2123

d. 0000:6727

e. FFFF:4336

f. 1080:0100

g. AB01:FFFF

What are the first and the last physical memory addresses accessible using the following segment values?

a. 1000

b. 0FFF

c. 0001

d. F090

- What physical address is accessed with [BX+SI] if FFFF is loaded in BX, SI, and DS.

Assuming the instruction in left column form a complete program, write the value of register AX after each instruction. Also assume that each instruction is exactly 3 bytes long in machine code.

| Instruction | AX Value (hex) |
|---|---|
| `[org 100h]` | --- |
| `Mov ax, myvalue` | |
| `Mov ax, 0` | --- |
| `Mov al, [mylist+5]` | |
| `Mov ax, 0` | --- |
| `Mov ah, [myvalue+1]` | |
| `Mov ax, 0` | --- |
| `Mov ax, [myvalue]` | |
| `Mov ax, 4c00h`<br>`int 21h`<br>`mylist:  dw 20h, 41h, 36h, 1Ah`<br>`myvalue: db 5h, 9h` | --- |

What is the **effective address** generated by the following combinations if they are valid. If not give reason.

Initially BX=0x0100, SI=0x0010, DI=0x0001, BP=0x0200, and SP=0xFFFF

a. bx-si

b. bx-bp

c. bx+10

d. bx-10

e. bx+sp

f. bx+di

Identify the problems in the following instructions and correct them by replacing them with one or two instruction having the same effect.

a. mov [02], [22]

b. mov [wordvar], 20

c. mov bx, al

d. mov ax, [si+di+100]

Identify which of the following instruction assemble correctly, which ones give warning and which ones will give error.

```
a.mov num1, 1
b.mov [num1], 1
c.mov num1, 0A0Bh
d.mov ax, [num1]
e.mov byte [num1], 0A0Bh
f.mov byte [num1], 0Ah
g.mov word [num1], 0A0Bh
h.mov ax, [ss:bx]
i.mov bx, [ax]
j.mov ax, [bx-10]
```

# If following is the listing file of code, then what will be the size of .com file?

```
 1                                          [org 0x0100]
 2
 3 00000000 A1[1000]                            mov ax, [num1]
 4 00000003 8A1E[1200]                          mov bl, [num2]
 5 00000007 B700                                mov bh,0
 6 00000009 01D8                                add ax, bx
 7
 8
 9 0000000B B8004C                          mov ax, 0x4c00 ; terminate program
10 0000000E CD21                            int 0x21
11
12 00000010 0201                            num1: dw 0102h
13 00000012 03                              num2: db 03h
14 00000013 0000                            sum: dw 0
```

# Identify the problem in following code.

```
[org 0x100]
   mov ax, [num1]
   mov bl, [num2]
   add ax, bx
   mov [sum], ax

    mov ax, 0x4c00 ; terminate program
    int 0x21

num1: dw 1,
num2: db 10
sum: dw 0
```

If AX=8FFF and BX=0FFF and “cmp ax, bx” is executed, which of the following jumps will be taken? Also give the value of Z, S, and C flags.

a. jg greater

b. jl smaller

c. ja above

d. jb below

- Give the value of the AX register and the carry flag after each of the following instructions.

```
stc
mov ax, 5204h
adc ah, 75h
cmc ; complement carry flag
xor ah, al
mov cl, 4
shr al, cl
rcr ah, cl
```

- Write code to swap every pair of bits in the AH register.

- Write code to swap every pair of bits in the AH register.

A possible algorithm

| | |
|---|---|
| AH | 10 11 01 00 |
| 1. Extract left bits of pairs | 10 10 00 00 |
| 2. Extract right bits of pairs | 00 01 01 00 |
| 3. Shift the result of step 1 right | 01 01 00 00 |
| 4. Shift the result of step 2 left | 00 10 10 00 |
| 5. Merge results of step 3 & 4 | 01 11 10 00 |

- Write code to calculate the number of one bits in BX register

- Write code to calculate the number of one bits in BX register

BX `0100 1011 1100 0101`

A possible algorithm

Initialize count = 0
Repeat 16 times:
    Shift BX right
    If CF = 1, count++