Assignment no 5 on ch 4 - lec 4.

Data Structures (National University of Computer and Emerging Sciences)

Assignment no: 5 on chapter no-4

1) **Write an assembly language program to perform word by byte division of two unsigned number.**

Ans: Program for word by byte division.

```
DATA SEGMENT
NUMBER1 DW 4359H
NUMBER2 DB 99H
Quotient DB 1 DUP(0)
Remainder DB 1 DUP(0)
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE, DS: DATA
START:MOV DX , DATA
        MOV DS ,DX
        MOV AX ,NUMBER1
        MOV BL ,NUMBER2
        DIV BL                          ; Ans AH :Quotient ,AL :Remainder in AX
        MOV Quotient, AL
        MOV Remainder, AH
        MOV AH , 4CH
        INT 21H
CODE ENDS
END START
```

2) **Write 8086 assembly language instruction for the following:**
(i)Move 5000H to register D
(ii)Multiply AL by 05H

Ans: (i) Move 5000H to register D

```
MOV DX, 5000H
```

(ii)Multiply AL by 05

```
MOV BL, 05H
MUL BL
```

3) **Write an ALP to perform addition of two 16 bit BCD number.**

Ans:

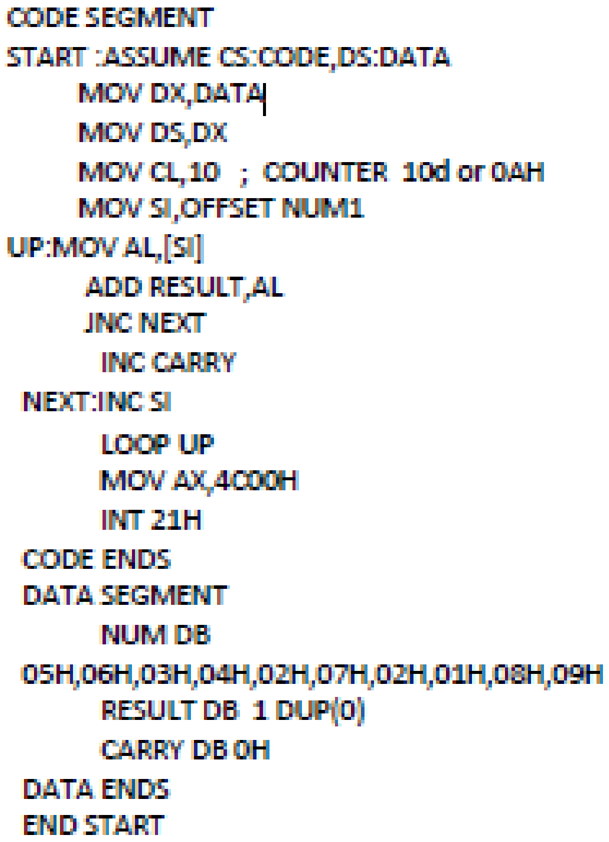
```
 DATA SEGMENT
N1 DW 2804H
N2 DW 4213H
BCD_SUM DW ?
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE, DS:DATA
START: MOV AX, DATA
        MOV DS, AX
        MOV AX, N1
        MOV BX, N2
        ADD AL,BL
        DAA                         ; LOWER BYTE ADDITION
        MOV CL,AL
        MOV AL,AH
        ADD AL,BH
        DAA                         ; HIGHER BYTE ADDITION
        MOV CH,AL
        MOV BCD_SUM, CX
        MOV AH,4CH
        INT 21H
CODE ENDS
END START
```

4) Write an ALP to count number of 1's in register DL.

Ans:

```
 DATA SEGMENT
N DB 43H
COUNT DB 0H
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE, DS: DATA
START:MOV DX , DATA
        MOV DS ,DX
        MOV DL ,N
        MOV CX ,08H
UP: SHR DL,01                ; any other shift/rotate instruction is also correct
        JNC NEXT
        INC COUNT
NEXT: LOOP UP
        MOV AH , 4CH
        INT 21H
CODE ENDS
END START
```

5) **Write an ALP to find sum of 10 numbers. (Assume numbers as 8 bits)**
Ans:



```
CODE SEGMENT
START :ASSUME CS:CODE,DS:DATA
        MOV DX,DATA
        MOV DS,DX
        MOV CL,10   ;  COUNTER  10d or 0AH
        MOV SI,OFFSET NUM1
UP:MOV AL,[SI]
         ADD RESULT,AL
         JNC NEXT
           INC CARRY
  NEXT:INC SI
           LOOP UP
           MOV AX,4C00H
           INT 21H
  CODE ENDS
  DATA SEGMENT
           NUM DB
  05H,06H,03H,04H,02H,07H,02H,01H,08H,09H
           RESULT DB  1 DUP(0)
           CARRY DB 0H
  DATA ENDS
  END START
```

6) **Write an ALP to arrange five 8 bit numbers in ascending order.**

```
Ans:    Data segment ; start of data segment
        Array db 15h,05h,08h,78h,56h
        Data ends                          ; end of data segment
        Code segment                       ; start of code segment
        Start: assume cs: code, ds:data
        mov dx, data                       ; initialize data segment
        mov ds, dx
        mov bl,05h                          ; initialize pass counter to read numbers from array
        step1: mov si,offset array          ; initialize memory pointer to read number
        mov cl,04h                         ; initialize byte counter

        step: mov al,[si]
        cmp al,[si+1]                      ; compare two numbers
        jc down                             ; if number <next no. Then go to down
        xchg al,[si+1]                      ; interchange numbers
        xchg al,[si]
        Down : add si,1                     ; increment memory pointer to point next
        loop step                           ;decrement byte counter if count is ? 0
        then step
        dec bl                              ; decrement pass counter if ? 0 then step1
        jnz step1
        Code ends
        End start
```

7) **Write an ALP to convert BCD to HEX.**

Ans:

```
ATA SEGMENT
DEC_NUM DB 56
HEX_NUM DW 0
MULT_FAC DW 3e8H
DIGIT_COUNT DW 2
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:MOV AX,DATA
MOV DS,AX
MOV BX,0AH
MOV CX,DIGIT_COUNT
MOV SI,OFFSET DEC_NUM
UP: MOV AL,[SI]
AND AX,000FH
MUL MULT_FAC
ADD HEX_NUM,AX
MOV AX,MULT_FAC
MOV DX,00
DIV BX
MOV MULT_FAC,AX
INC SI
LOOP UP
ENDS
END START
```

8) **Write an ALP to reserve a string of 8 characters.**

Ans:

```
Data segment
 string db ‘goodmorn’
 rev db 0fh dup(?)
 Data ends
 Code segment
 assume cs:code, ds:data
 start: mov dx, data
            mov ds,dx
             lea si, string
             mov cx,0fh
             lea di, rev
             add di,0fh
             up: mov al, [si]
             mov [di], al
             Inc si
             dec di
             loop up
             code ends
             end start
```

9) **Write an ALP to count odd numbers in an array of five 8 bit numbers.**

Ans:

```
DATA SEGMENT
ARRAY DB 02H,05H,06H,07H,03H
ODD DB 00H
DATA ENDS
CODE SEGMENT
START:ASSUME CS:CODE,DS:DATA
        MOV DX,DATA
        MOV DS,DX
        MOV CL,05H
        MOV SI,OFFSET ARRAY
NEXT:MOV AL,[SI]
        ROR AL,1                ;or RCR
        JNC DN                  ; Check for Odd
        INC ODD
DN : INC SI
LOOP NEXT
        MOV AH,4CH
        INT 21H
CODE ENDS
END START
```

10) **Write assembly language instruction of 8086 microprocessor to :**
i. Copy 1000H to register BX

ii. Rotate register BL left four times
Ans:

```
i) MOV BX, 1000H

ii) MOV CL, 04H
RCL BL, CL
Or
MOV CL, 04H
ROL BL, CL
```

11) **Write an assembly language program to add two BCD numbers.**

Ans:

```
DATA SEGMENT
NUM1 DB 09H
NUM2 DB 09H
SUM DB ?
DATA ENDS
CODE SEGMENT
START: ASSUME CS:CODE,DS:DATA
          MOV AX,DATA
          MOV DS,AX
          MOV AL,NUM1
          ADD AL,NUM2
          DAA                         ;Decimal adjust for addition
          MOV SUM,AL
          MOV AH,4CH
          INT 21H
          CODE ENDS
```

-or-

```
.MODEL SMALL
.DATA
NUM1 DB 84H
NUM2 DB 28H
RES_LSB DB ?
RES_MSB DB ?
.CODE
MOV AX,@DATA
MOV DS,AX
MOV AL,NUM1
MOV BL,NUM2
ADD AL,BL                             ;Ans ACH
DAA
JNC DN
INC RES_MSB
DN:MOV RES_LSB,AL
MOV AH,4CH
INT 21H
END
```

12) **Write an ALP to count of zero’s in BL register.**

```
Ans:    DATA SEGMENT
        NUM DB 0F3H ;BINARY{ 1111 0011}
        ZEROS DB 0
        DATA ENDS
        CODE SEGMENT
        START: ASSUME CS:CODE,DS:DATA
        MOV AX,DATA
        MOV DS,AX
        MOV CX,8                      ;rotation counter
        MOV BL,NUM
        UP:
        ROR BL,1                      ; RCR,ROL , RCL can be used
        JC DN ;IF CARRY loop
        INC ZEROS                     ; else increment 0's count ;ANSWER 02
        DN:LOOP UP                    ;decrement rotation counter
        EXIT: MOV AH,4CH
        INT 21H
        CODE ENDS
        END START
```

13) **Write an ALP to subtract two 8 bit numbers.**

```
Ans:     DATA SEGMENT
        NUM1 DB 10H
        NUM2 DB 20H
        DIFF DB ?
        DATA ENDS
        CODE SEGMENT
        START: ASSUME CS:CODE,DS:DATA
                MOV AX,DATA
                MOV DS,AX
                MOV AL,NUM1

                MOV BL,NUM2
                SUB AL,BL
                MOV DIFF,AL
                MOV AH,4CH
                INT 21H
        CODE ENDS
        END START
```

-or-

```
        DATA SEGMENT
        NUM1 DB 85H
        NUM2 DB 92H
        DIFFERENCE DB 1 DUP(0)
        DATA ENDS
        CODE SEGMENT
        ASSUME CS:CODE,DS:DATA
        START:
                MOV DX,DATA
                MOV DS,DX
                MOV AL,NUM1
                MOV BL,NUM2
                SUB AL,BL
                MOV DIFFERENCE,AL
        JNC EXIT
                MOV DIFFERENCE+1,01
                EXIT:MOV AH,4CH
                INT 21H
        CODE ENDS
        END START
```

14) **Write an ALP to add two 16 bit numbers.**

Ans:

```
DATA SEGMENT
NUMBER1 DW 5522 H
NUMBER2 DW 3311H
SUM DW 2 DUP(0)
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
        MOV DX,DATA
        MOV DS,DX
        MOV AX,NUMBER1
        MOV BX,NUMBER2
        ADD AX,BX
        MOV SUM,AX
        MOV AH,4CH

        INT 21H

CODE ENDS
END START
```

-Or-

```
DATA SEGMENT
NUMBER1 DW 5522 H
NUMBER2 DW 8311H
SUM DW 2 DUP(0)
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
        MOV DX,DATA
        MOV DS,DX
        MOV AX,NUMBER1
        MOV BX,NUMBER2
        ADD AX,BX
        MOV SUM,AX
        JNC EXIT                ;EXIT IF CARRY
        MOV SUM+2,01            ;STORE CARRY BIT IN MS DIGIT
EXIT:MOV AH,4CH
        INT 21H
CODE ENDS
END START
```

15) **Write an ALP to find sum of series 0BH, 05H, 07H, 0AH,01H.**

Ans: **DATA SEGMENT**

```
NUM1 DB 0BH,05H,07H,0AH,01H

RESULT DB 1 DUP(0)
CARRY DB 0H
DATA ENDS
CODE SEGMENT
START:ASSUME CS:CODE,DS:DATA
MOV DX,DATA
MOV DS,DX
MOV CL,05H
MOV SI,OFFSET NUM1
UP:MOV AL,[SI]
        ADD RESULT,AL               ;Answer : AL : 22H
        JNC NEXT
        INC CARRY
NEXT:INC SI
LOOP UP
        MOV AX,4C00H
        INT 21H
CODE ENDS
END START
```

16) Write ALP to compute, whether the number in BL register is even or odd.

Ans: DATA SEGMENT

```
NUM DB 9H
ODD DB 0
EVEN_NO DB 0
DATA ENDS
CODE SEGMENT
START: ASSUME CS:CODE,DS:DATA
        MOV AX,DATA
        MOV DS,AX
        MOV BL,NUM
        ROR BL,1                    ;or RCR
        JNC DN                       ; check ENEN or ODD
        ROL BL,1                    ; restore number
        MOV ODD,BL                   ; odd
        JMP EXIT
DN: ROL BL,1
        MOV EVEN_NO,BL              ; even no
EXIT: MOV AH,4CH
        INT 21H
CODE ENDS
END START
```

17) **Write an ALP to transfer 10 bytes of data from one memory location to another Also draw the flow chart for the same.**

Ans:
```
DATA SEGMENT
block1 db 10 dup(10h)
block2 db 10 dup(0)
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA ,ES: EXTRA
START:MOV DX,DATA                              ;initialize data seg
        MOV DS,DX
        MOV DX, EXTRA
        MOV ES,DX
        LEA SI,BLOCK1
        LEA DI,BLOCK2
        MOV CX,000AH
        CLD
        REP MOVSB
        MOV AH,4CH
        INT 21H
CODE ENDS
END START
```

-or-

```
DATA SEGMENT
block1 db 10 dup(10h)
block2 db 10 dup(0)
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:MOV DX,DATA                        ;initialize data seg
        MOV DS,DX
        MOV ES,DX
        LEA SI,BLOCK1
        LEA DI,BLOCK2
        MOV CX,000AH
        CLD
BACK:MOV AL,[SI]                          ; REP MOVSB
        MOV [DI],AL
        INC SI
        INC DI
        DEC CX
        JNZ BACK
        MOV AH,4CH
        INT 21H
CODE ENDS
END START
```

18) **Write an ALP to transfer a block of 10 data bytes using string instruction.**

Ans:

```
DATA SEGMENT
STRNO1 DB 10 DUP(10) (Any Value in STRNO1 should be given correct)
DATA ENDS
EXTRA SEGMENT
STRNO2 DB 10 DUP(0)
EXTRA ENDS
CODE SEGMENT
ASSUME CS: CODE, DS: DATA, ES: EXTRA
START:
        MOV DX, DATA
        MOV DS, DX
        MOV DX, EXTRA
        MOV ES, DX

        LEA SI, STRNO1
        LEA DI, STRNO2
        MOV CX, 000AH
        CLD
        REP MOVSB
        MOV AH, 4CH
        INT 21H
CODE ENDS
END START
```

19) **Write an ALP for BCD to hex conversion.**

Ans: DATA SEGMENT

```
BCD DB 56D
HEX DB ?
DATA ENDS
CODE SEGMENT

ASSUME CS:CODE, DS:DATA
START:
            MOV AX,DATA
            MOV DS,AX
            MOV AL,BCD
            MOV AH,00H
            MOV BL,10H
            DIV BL
            MOV DL,AH
            MOV AH,00H
            DIV BL
            MOV CL,04H
            ROR AH,CL
            OR DL,AH
            MOV HEX,DL
            MOV AH,4CH
            INT 21H
CODE ENDS
END START
```

-or-

```
                         DATA SEGMENT

DEC_NUM DB 56
HEX_NUM DW 0
MULT_FAC DW 3e8H
DIGIT_COUNT DW 2
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:MOV AX,DATA
            MOV DS,AX
            MOV BX,0AH
            MOV CX,DIGIT_COUNT
            MOV SI,OFFSET DEC_NUM
UP: MOV AL,[SI]
            AND AX,000FH
            MUL MULT_FAC
            ADD HEX_NUM,AX
            MOV AX,MULT_FAC
            MOV DX,00
            DIV BX
            MOV MULT_FAC,AX
            INC SI
```

```
LOOP UP
ENDS

END START
```

20) **Write assembly language program to divide two 16 bit unsigned numbers.**

Ans: Note: Since 8086 Performs 32bit / 16 bit division or 16bit / 8 bit division therefore for Two 16bit Number division we have to perform 32bit / 16bit Division.

Program for Double word by word division.

```
DATA SEGMENT
NUMBER1 DD00004359H
NUMBER2 DW1199H
Quotient DW 1 DUP(0)
Remainder DW 1 DUP(0)
DATA ENDS

CODE SEGMENT
ASSUME CS: CODE, DS: DATA
START:
        MOV DX , DATA
        MOV DS ,DX
        LEA SI, NUMBER1                 ; Moving 32 Bit Number into DX : AX
        MOV AX , [SI]
        INC SI
        INC SI
        MOV DX, [SI]
        MOV BX ,NUMBER2                  ; Moving 16 Bit Number into BX
        DIV BX                          ; Ans = AX:Quotient,DX :Remainder
        MOV Quotient, AX
        MOV Remainder, DX
        MOV AH , 4CH
        INT 21H
CODE ENDS
END START
```

21) **Write an assembly language program to multiply two 8 bit number.**

Ans:
```
DATA SEGMENT
NUM1 DB 05H
NUM2 DB 02H
RESULT DW?
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA

START:
        MOV DX,DATA
        MOV DS,DX
        MOV AL,NUM1
        MOV AH,NUM2
        MUL AH
        MOV RESULT,AX
CODE ENDS
END START
```

22) **Write an assembly language program to add the series of 5 number**

Ans:
```
DATA SEGMENT
NUM1 DB10H,20H,30H,40H,50H
RESULT DB 1 DUP(0)
CARRY DB 0H

DATA ENDS
CODE SEGMENT:
ASSUMECS:CODE,DS:DATA
START:
        MOV DX,DATA
        MOV DS,DX
        MOV CL,05H
        MOV SI,OFFSET NUM1                          OR  LEA SI, NUM1
        UP:MOV AL,[SI]
        ADD RESULT,AL
JNC NEXT
        INC CARRY
NEXT: INC SI
LOOP UP                                   OR  DEC CL,
        JNZ UP
        MOV AH,4C00H
        INT 21H
CODE ENDS
END START
```

23) **Write an ALP to count the number of '1' in a 16 bit number. Assume the number to be stored in BX register. Store the result in CX register.**

Ans:

```
DATA SEGMENT
NUM DW 0FF33H ;BINARY{ 1111 1111 0011 0011}
ONES DB 0
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
START:
        MOV AX,DATA
        MOV DS,AX
        MOV CX,16                   OR MOV CX, 10H ;rotation counter
        MOV AX,NUM
UP      :ROR AX,1
        JNC DN                      ;IF no CARRY loop
        INC ONES                    ; else increment1's count
DN:LOOP UP                          OR DEC CX ;decrement rotation counter
        JNZ UP
        MOV CX,ONES
        MOV AH,4CH
        INT 21H
CODE ENDS
END START
```

24) **How many times LOOP1 will be executed in following program? What will be the contents of BL after the execution?**

MOV BL, 00H
MOV CL, 05H
LOOP1 : ADD BL, 02H
DEC CL
JNZ LOOP1

Ans: LOOP1 will be executed 5 times in the above program
The contents of BL will be 0Ah after the execution of program.

25) **Write an ALP to add l6 bit BCD number.**

Ans:
```
DATA SEGMENT
N1 DW 2804H
N2 DW 4213H
BCD_SUM DW ?
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE, DS:DATA
START:
        MOV AX, DATA
        MOV DS, AX
        MOV AX, N1
        MOV BX, N2
        ADD AL,BL
        DAA                          ; LOWER BYTE ADDITION
        MOV CL,AL
        MOV AL,AH
        ADD AL,BH
        DAA                          ; HIGHER BYTE ADDITION
        MOV CH,AL
        MOV BCD_SUM, CX
        MOV AH,4CH
        INT 21H
CODE ENDS
END START
```

26) **Rotate content of AX by 4 bit towards left.**

Ans: Rotate content of AX by 4 bit towards left.
```
MOV CL, 04H
RCL AX, CL
```
Or
```
MOV CL, 04H
ROL AX, CL
```

27) **Write ALP to divide two 16 bit numbers.**

Ans: DATA SEGMENT

```
A DW 4444H
 B DW 0002H
C DW ?
DATA ENDS
CODE SEGMENT
ASSUME DS:DATA, CS:CODE
START:
        MOV AX,DATA
        MOV DS,AX
        MOV AX,A
        MOV BX,B
        DIV BX
        MOV C,AX
        INT 3
CODE ENDS
END START
```

28) Describe any two bit manipulation instructions

Ans: • AND – Used for ANDing each bit in a source operand with the corresponding bit in destination operand byte/word. And the result is stored in Destination operand.

Eg:

AND BH, AL; AND bit by bit Byte in AL with data in BH and the result is stored in BH

• **OR** – Used to multiply each bit in a byte/word with the corresponding bit in another byte/word.
Eg:
OR AX, 00ABH; OR bit by bit word in AX with immediate data 00ABH
and the result is stored in AX
Eg:
• **XOR** – Used to perform Exclusive-OR operation over each bit in a byte/word
with the corresponding bit in another byte/word.
Eg:
XOR CX, [SI]; XOR bit by bit word at offset [SI] in DS with word in CX
and the result is stored in CX
Eg:
• **NOT** – Used to invert each bit of a byte or word.
Eg:
NOT AX; Complement the contents of AX

29) **Write an ALP to find largest number from array of 10 numbers**

Ans: DATA SEGMENT

```
ARRAY DB 15H,45H,08H,78H,56H,02H,04H,12H,23H,09H

LARGEST DB 00H
DATA ENDS
CODE SEGMENT
START:ASSUME CS:CODE,DS:DATA
        MOV DX,DATA
        MOV DS,DX
        MOV CX,09H
        MOV SI ,OFFSET ARRAY
        MOV AL,[SI]
UP:INC SI
        CMP AL,[SI]
JNC NEXT ;CHANGE
        MOV AL,[SI]
NEXT:DEC CX
JNZ UP
        MOV LARGEST,AL                  ; AL=78h
        MOV AX,4C00H
        INT 21H
CODE ENDS
END START
```

30) **Write an ALP to find length of string**

Ans: DATA SEGMENT

```
STR1 DB 'STUDENT$'

LENGTH_STRING DB?
DATA ENDS
ASSUME CS:CODE, DS:DATA
CODE SEGMENT
START: MOV AX, DATA
        MOV DS, AX
        MOV AL, '$'
        MOV CX, 00H
        MOV SI, OFFSET STR1
BACK: CMP AL, [SI]
        JE DOWN
        INC CL
        INC SI
        JMP BACK
DOWN: MOV LENGTH_STRING, CL
        MOV AX, 4C00H
        INT 21H
CODE ENDS
END
```