FR. Conceicao Rodrigues College of Engineering
Department of Computer Engineering

1. Addition of Two 8/16/32 bit numbers

1. Course, Subject & Experiment Details

Academic Year	**2023-24**	**Estimated Time**	**Experiment No. 1– 02 Hours**
Course & Semester	**S.E. (Comps) – Sem. IV**	**Subject Name**	**Microprocessor**
Chapter No.	**2**	**Chapter Title**	**Instruction Set and Programming**
Experiment Type	**Software**	**Subject Code**	**CSC405**

Rubrics

Timeline **(2)**	**Practical Skill & Applied Knowledge** **(2)**	**Output** **(3)**	**Postlab** **(3)**	**Total (10)**	**Sign**

2. Aim & Objective of Experiment

TO ADD TWO 8/16/32 BIT NUMBERS

Objective : Program involves storing the two 8-bit no in memory locations and adding them taking into consideration the carry generated. The objective of this program is to give an overview of arithmetic instructions of 8086 for 8-bit operands

3. Software Required

TASM Assembler

Prepared by : Prof. Heenakausar Pendhari

4 . Brief Theoretical Description

Pre-Requisites: 1. Instructions of microprocessor 8086
2. Addressing mode of microprocessor 8086.
3. Knowledge of TASM directories.

Theory: The addressing modes used in program are:

1) Direct addressing mode: in this mode address of operand is directly specified in the instruction. This address is offset address of the segment being indicated by an instruction.

   E.g. MOV AL,[2000h]
   EA = DS x 10H + 2000H

2) Register Addressing Mode: In this mode operand are specified using registers. Instructions are shorter but operations cannot be identified looking at instruction.

   E.g. MOV CL, DL

3) Based Indexed Addressing Mode: The operand address is calculated using base register and index register.

   E.g. MOV DX, [BX + SI] moves word from address pointed by BX + SI in data segment to DX.

   EA = DS x 10H + BX + SI

4) Base indexed plus displacement: In this mode address of operand is calculated using base register , index register and displacement.

   E.g. MOV CX, [BX+DI+10h]

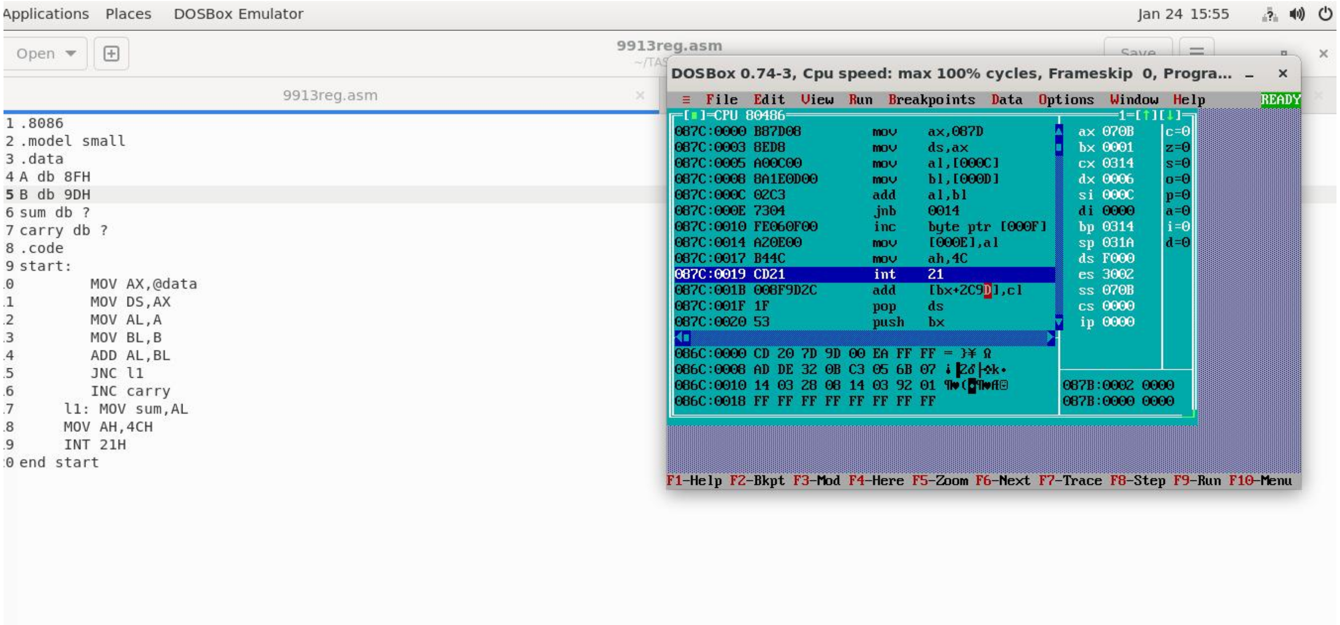
   This moves a word from address pointed by BX + DI +10h of segment to CX.

5. Algorithm:

1. Initialize the data segment.
2. Store two 8/16 -bit numbers in memory locations.
3. Move the 1st number in any one of the general purpose register.
4. Move the 2nd number in any other general purpose register.
5. Add the 2 numbers.
6. Store the result in memory location.
7. Check for carry flag. If carry flag is set then store '1' as MSB of result.
8. Stop

6. Conclusion:

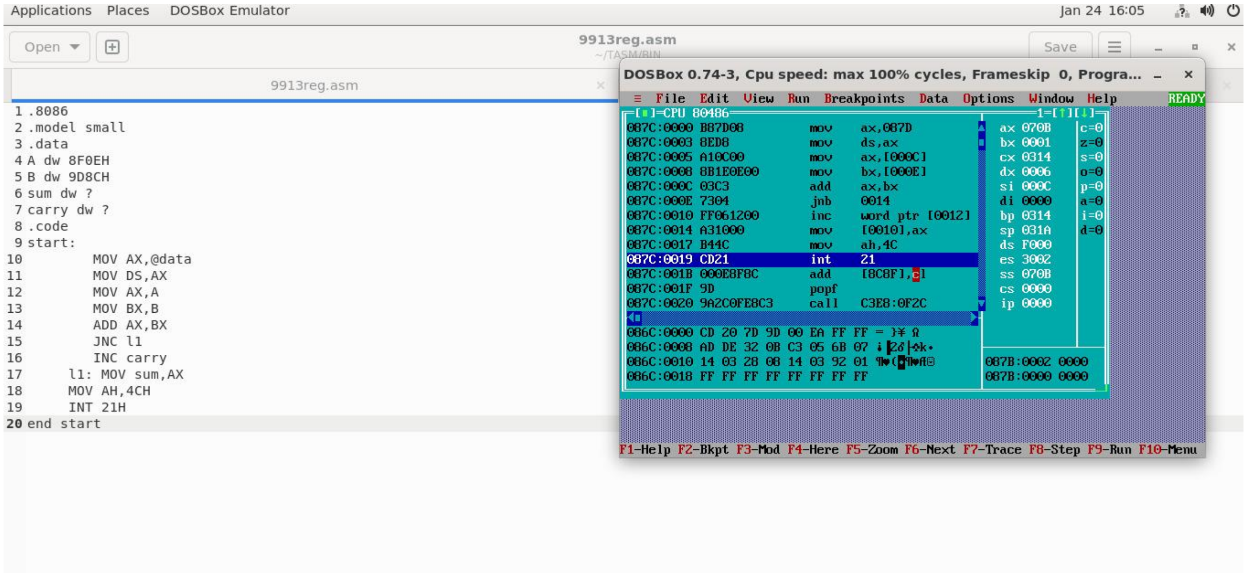
8 bit addition:-


Applications
Places
DOSBox Emulator
Jan 24 15:55
Open
9913reg.asm
~/TAS
Save
9913reg.asm
DOSBox 0.74-3, Cpu speed: max 100% cycles, Frameskip 0, Progra...
≡ File Edit View Run Breakpoints Data Options Window Help
READY
F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

```
.8086
.model small
.data
A db 8FH
B db 9DH
sum db ?
carry db ?
.code
start:
        MOV AX,@data
        MOV DS,AX
        MOV AL,A
        MOV BL,B
        ADD AL,BL
        JNC l1
        INC carry
     l1: MOV sum,AL
     MOV AH,4CH
     INT 21H
end start
```

```
[■]=CPU 80486                                          1=[↑][↓]
087C:0000 B87D08        mov     ax,087D                ax 070B   c=0
087C:0003 8ED8          mov     ds,ax                  bx 0001   z=0
087C:0005 A00C00        mov     al,[000C]              cx 0314   s=0
087C:0008 8A1E0D00      mov     bl,[000D]              dx 0006   o=0
087C:000C 02C3          add     al,bl                  si 000C   p=0
087C:000E 7304          jnb     0014                   di 0000   a=0
087C:0010 FE060F00      inc     byte ptr [000F]        bp 0314   i=0
087C:0014 A20E00        mov     [000E],al              sp 031A   d=0
087C:0017 B44C          mov     ah,4C                  ds F000
087C:0019 CD21          int     21                     es 3002
087C:001B 008F9D2C      add     [bx+2C9D],cl           ss 070B
087C:001F 1F            pop     ds                     cs 0000
087C:0020 53            push    bx                     ip 0000

086C:0000 CD 20 7D 9D 00 EA FF FF = }¥ Ω
086C:0008 AD DE 32 0B C3 05 6B 07 ¡▌2♂├♣k•              087B:0002 0000
086C:0010 14 03 28 08 14 03 92 01 ¶♥(◘¶♥Æ☺              087B:0000 0000
086C:0018 FF FF FF FF FF FF FF FF
```

16 bit addition:-


Applications Places DOSBox Emulator
Jan 24 16:05
Open
9913reg.asm
~/TASM/BIN
Save
9913reg.asm
DOSBox 0.74-3, Cpu speed: max 100% cycles, Frameskip 0, Progra...
≡ File Edit View Run Breakpoints Data Options Window Help
READY

```
.8086
.model small
.data
A dw 8F0EH
B dw 9D8CH
sum dw ?
carry dw ?
.code
start:
        MOV AX,@data
        MOV DS,AX
        MOV AX,A
        MOV BX,B
        ADD AX,BX
        JNC l1
        INC carry
     l1: MOV sum,AX
     MOV AH,4CH
     INT 21H
end start
```

```
[■]=CPU 80486                                        1=[↑][↓]
087C:0000 B87D08       mov    ax,087D              ax 070B   c=0
087C:0003 8ED8         mov    ds,ax                bx 0001   z=0
087C:0005 A10C00       mov    ax,[000C]            cx 0314   s=0
087C:0008 8B1E0E00     mov    bx,[000E]            dx 0006   o=0
087C:000C 03C3         add    ax,bx                si 000C   p=0
087C:000E 7304         jnb    0014                 di 0000   a=0
087C:0010 FF061200     inc    word ptr [0012]      bp 0314   i=0
087C:0014 A31000       mov    [0010],ax            sp 031A   d=0
087C:0017 B44C         mov    ah,4C                ds F000
087C:0019 CD21         int    21                   es 3002
087C:001B 000E8F8C     add    [8C8F],cl            ss 070B
087C:001F 9D           popf                        cs 0000
087C:0020 9A2C0FE8C3   call   C3E8:0F2C            ip 0000

086C:0000 CD 20 7D 9D 00 EA FF FF = }¥ Ω
086C:0008 AD DE 32 0B C3 05 6B 07 ¡▌2δ├♠k•
086C:0010 14 03 28 08 14 03 92 01 ¶♥(◘¶♥Æ☺    087B:0002 0000
086C:0018 FF FF FF FF FF FF FF FF            087B:0000 0000

F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu
```

Postlab:

1. Write a program for addition of two 32 bit numbers ,execute and take the screen shots of the results.

.8086

.model small

.data

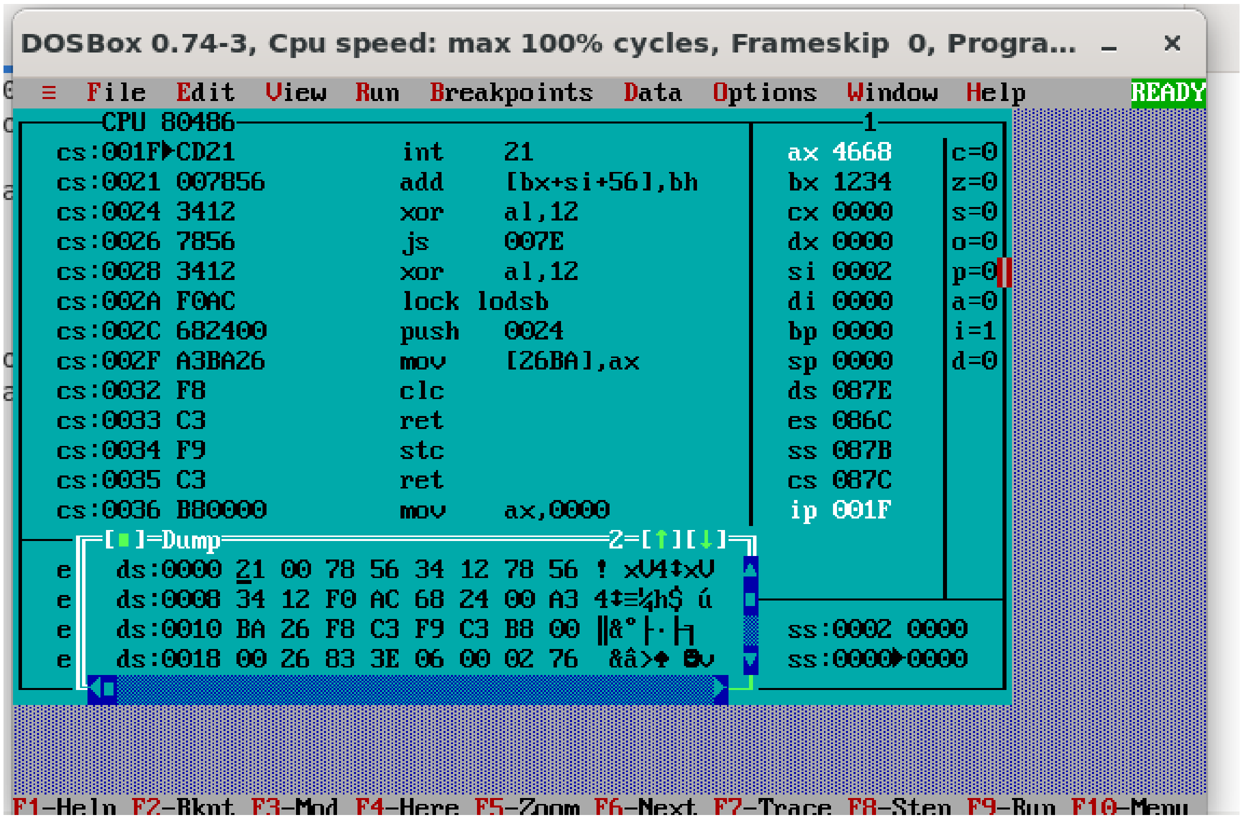
num1 dd 12345678H

num2 dd 12345678H

suml dw ?

Prepared by : Prof. Heenakausar Pendhari

```
    sumh dw ?
.code
start:
    MOV AX, @data
    MOV DS, AX
    LEA SI, num1
    MOV AX, [SI]
    MOV BX, [SI+04H]
    ADC AX, BX
    MOV suml, AX

    MOV AX, [SI+02H]
    MOV BX, [SI+06H]
    ADC AX, BX
    MOV sumh, AX
    MOV AH, 46H
    INT 21H
    end start
```



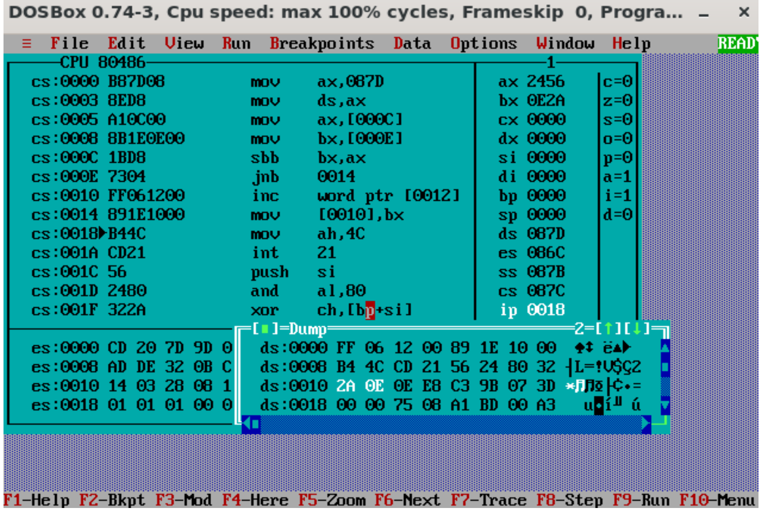
DOSBox 0.74-3, Cpu speed: max 100% cycles, Frameskip 0, Progra...
≡ File Edit View Run Breakpoints Data Options Window Help
READY
F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

```
CPU 80486                                              1
  cs:001F▸CD21         int    21                  ax 4668   c=0
  cs:0021 007856       add    [bx+si+56],bh       bx 1234   z=0
  cs:0024 3412         xor    al,12               cx 0000   s=0
  cs:0026 7856         js     007E                dx 0000   o=0
  cs:0028 3412         xor    al,12               si 0002   p=0
  cs:002A F0AC         lock lodsb                 di 0000   a=0
  cs:002C 682400       push   0024                bp 0000   i=1
  cs:002F A3BA26       mov    [26BA],ax           sp 0000   d=0
  cs:0032 F8           clc                        ds 087E
  cs:0033 C3           ret                        es 086C
  cs:0034 F9           stc                        ss 087B
  cs:0035 C3           ret                        cs 087C
  cs:0036 B80000       mov    ax,0000             ip 001F
[■]=Dump                                  2=[↑][↓]
  ds:0000 21 00 78 56 34 12 78 56 ! xV4‡xV
  ds:0008 34 12 F0 AC 68 24 00 A3 4‡≡¼h$ ú
  ds:0010 BA 26 F8 C3 F9 C3 B8 00 ║&°├·├╕
  ds:0018 00 26 83 3E 06 00 02 76  &â>♠ ☻v     ss:0002 0000
                                                ss:0000▸0000
```

2. Write a program to Subtract two 16 bit numbers.

```
.8086
.model small
.data
   A dw 2456H
   B dw 3280H
   subt dw ?
   burrow dw ?
.code
start:
```

```
MOV AX, @data

MOV DS, AX

MOV AX, A

MOV BX, B

SBB BX, AX

JNC skip

INC burrow

skip: MOV subt, BX

MOV AH, 4CH

INT 21H

end start
```


DOSBox 0.74-3, Cpu speed: max 100% cycles, Frameskip 0, Progra...
≡ File Edit View Run Breakpoints Data Options Window Help
READY
F1-Help F2-Bkpt F3-Mod F4-Here F5-Zoom F6-Next F7-Trace F8-Step F9-Run F10-Menu

```
CPU 80486                                                  1
  cs:0000 B87D08         mov    ax,087D             ax 2456   c=0
  cs:0003 8ED8           mov    ds,ax               bx 0E2A   z=0
  cs:0005 A10C00         mov    ax,[000C]           cx 0000   s=0
  cs:0008 8B1E0E00       mov    bx,[000E]           dx 0000   o=0
  cs:000C 1BD8           sbb    bx,ax               si 0000   p=0
  cs:000E 7304           jnb    0014                di 0000   a=1
  cs:0010 FF061200       inc    word ptr [0012]     bp 0000   i=1
  cs:0014 891E1000       mov    [0010],bx           sp 0000   d=0
  cs:0018▶B44C           mov    ah,4C               ds 087D
  cs:001A CD21           int    21                  es 086C
  cs:001C 56             push   si                  ss 087B
  cs:001D 2480           and    al,80               cs 087C
  cs:001F 322A           xor    ch,[bp+si]          ip 0018

  es:0000 CD 20 7D 9D 0    [■]=Dump                                  2=[↑][↓]
  es:0008 AD DE 32 0B C     ds:0000 FF 06 12 00 89 1E 10 00   ♠↕ ë▲▶
  es:0010 14 03 28 08 1     ds:0008 B4 4C CD 21 56 24 80 32  ┤L=!V$Ç2
  es:0018 01 01 01 00 0     ds:0010 2A 0E 0E E8 C3 9B 07 3D  *♫♫Φ├¢•=
                            ds:0018 00 00 75 08 A1 BD 00 A3    u◘í╜ ú
```