FR. Conceicao Rodrigues College of Engineering
Department of Computer Engineering

6. Matrix Addition/ Multiplication.

1. Course, Subject & Experiment Details

Academic Year	2023-24	Estimated Time	Experiment No. 6– 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

Perform Addition & Multiplication of 3 X 3 Matrix

Objective : The objective is to Add & multiply 3 X 3 matrix

3. Software Required

TASM Assembler

Prepared by : Prof. Heenakausar Pendhari

4 . Brief Theoretical Description

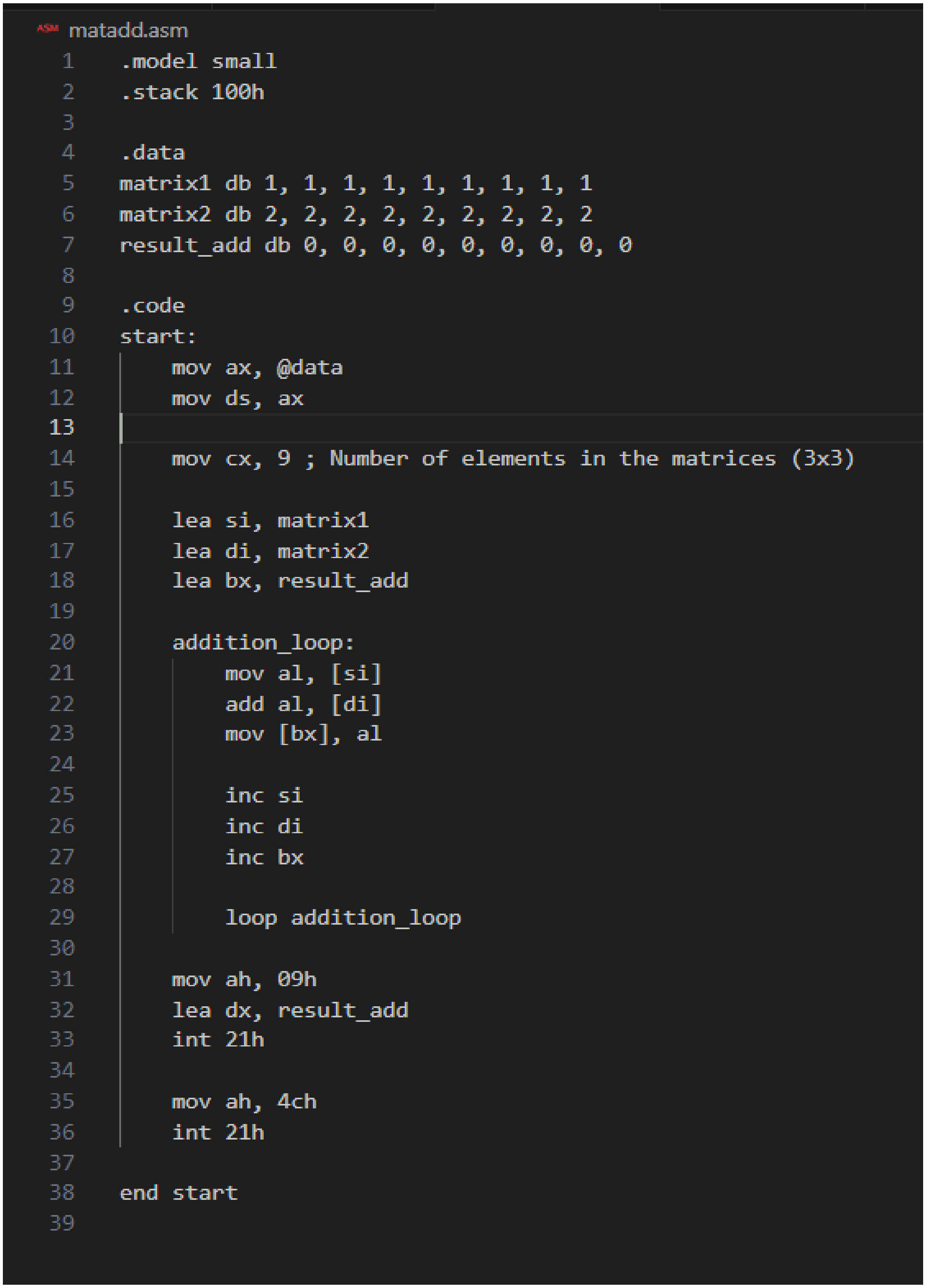
Pre-Requisites: 1. Knowledge of TASM directives.

2. Knowledge of DOS interrupts.
3. Knowledge of string instruction and MACRO

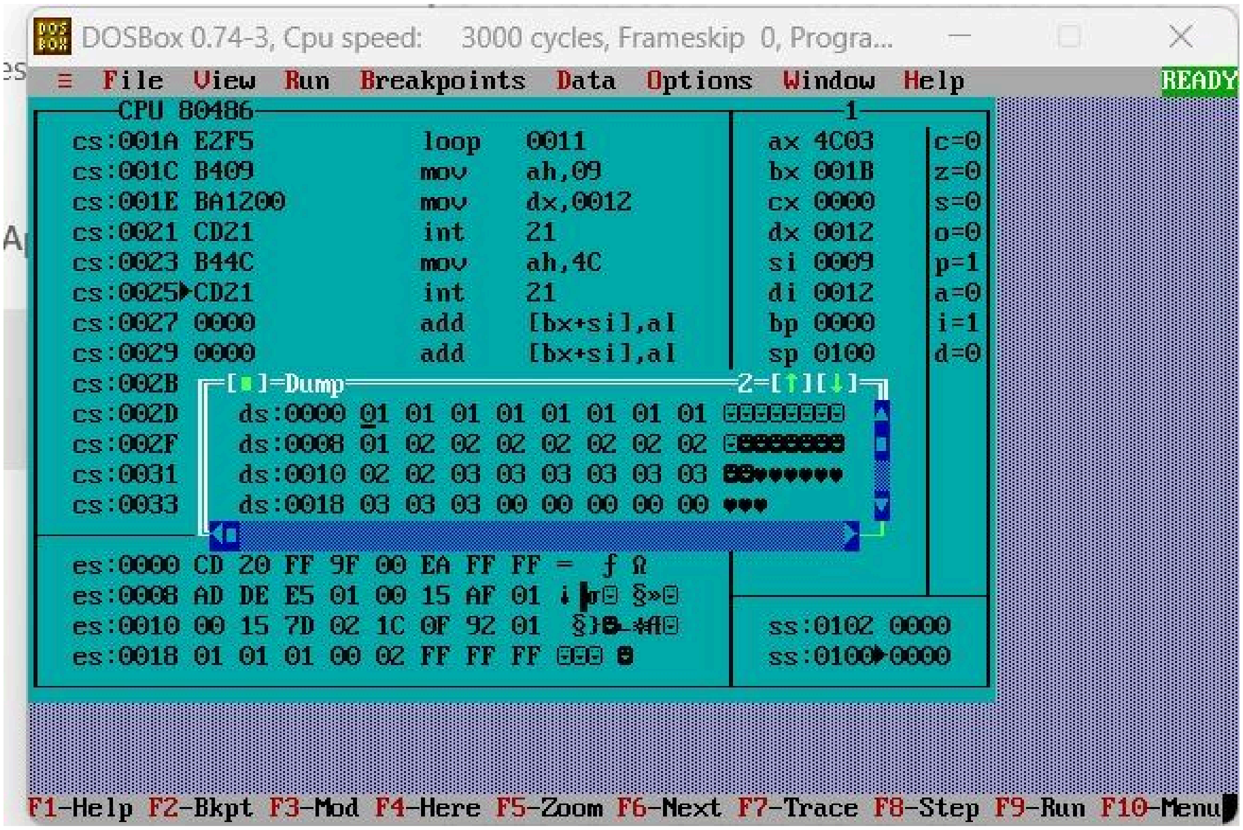
5. Algorithm:

1. Initialize the data segment.
2.Initialize counter = 9
3. Initialize pointer DI to matrix 1.
4. Initialize pointer BX to matrix 2.
5. Initialize pointer SI to result matrix 3.
6. Get the number from matrix 1.
7. Add number from matrix 1 with matrix 2 number.
8. Save the carry if any.
9. Save the result in result matrix 3.
10. Increment DI, BX, and SI to point to next element.
11. Decrement count.
12.Check if count = 0,if not go to step VI else go to step XIII
13. Display the result.
14. Stop.

Matrix addition:-



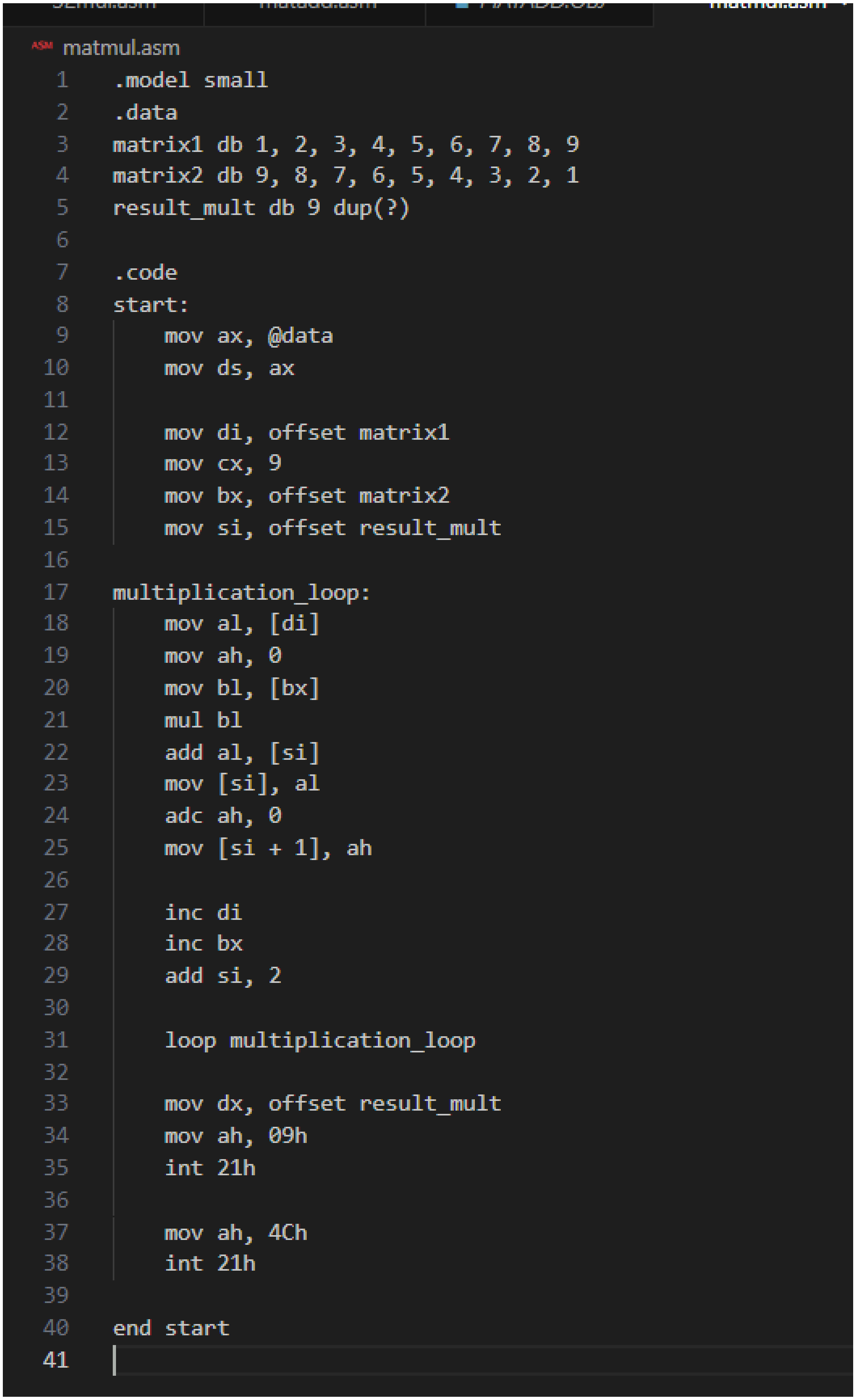
```
.model small
.stack 100h

.data
matrix1 db 1, 1, 1, 1, 1, 1, 1, 1, 1
matrix2 db 2, 2, 2, 2, 2, 2, 2, 2, 2
result_add db 0, 0, 0, 0, 0, 0, 0, 0, 0

.code
start:
    mov ax, @data
    mov ds, ax

    mov cx, 9 ; Number of elements in the matrices (3x3)

    lea si, matrix1
    lea di, matrix2
    lea bx, result_add

    addition_loop:
        mov al, [si]
        add al, [di]
        mov [bx], al

        inc si
        inc di
        inc bx

        loop addition_loop

    mov ah, 09h
    lea dx, result_add
    int 21h

    mov ah, 4ch
    int 21h

end start
```


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
≡ File 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:001A E2F5         loop   0011             ax 4C03   c=0
cs:001C B409         mov    ah,09            bx 001B   z=0
cs:001E BA1200       mov    dx,0012          cx 0000   s=0
cs:0021 CD21         int    21               dx 0012   o=0
cs:0023 B44C         mov    ah,4C            si 0009   p=1
cs:0025►CD21         int    21               di 0012   a=0
cs:0027 0000         add    [bx+si],al       bp 0000   i=1
cs:0029 0000         add    [bx+si],al       sp 0100   d=0
cs:002B    [■]=Dump===========================2=[↑][↓]
cs:002D     ds:0000 01 01 01 01 01 01 01 01
cs:002F     ds:0008 01 02 02 02 02 02 02 02
cs:0031     ds:0010 02 02 03 03 03 03 03 03 ♥♥♥♥♥♥
cs:0033     ds:0018 03 03 03 00 00 00 00 00 ♥♥♥

es:0000 CD 20 FF 9F 00 EA FF FF =  ƒ Ω
es:0008 AD DE E5 01 00 15 AF 01 ↓  §»
es:0010 00 15 7D 02 1C 0F 92 01  §}        ss:0102 0000
es:0018 01 01 01 00 02 FF FF FF            ss:0100►0000
```

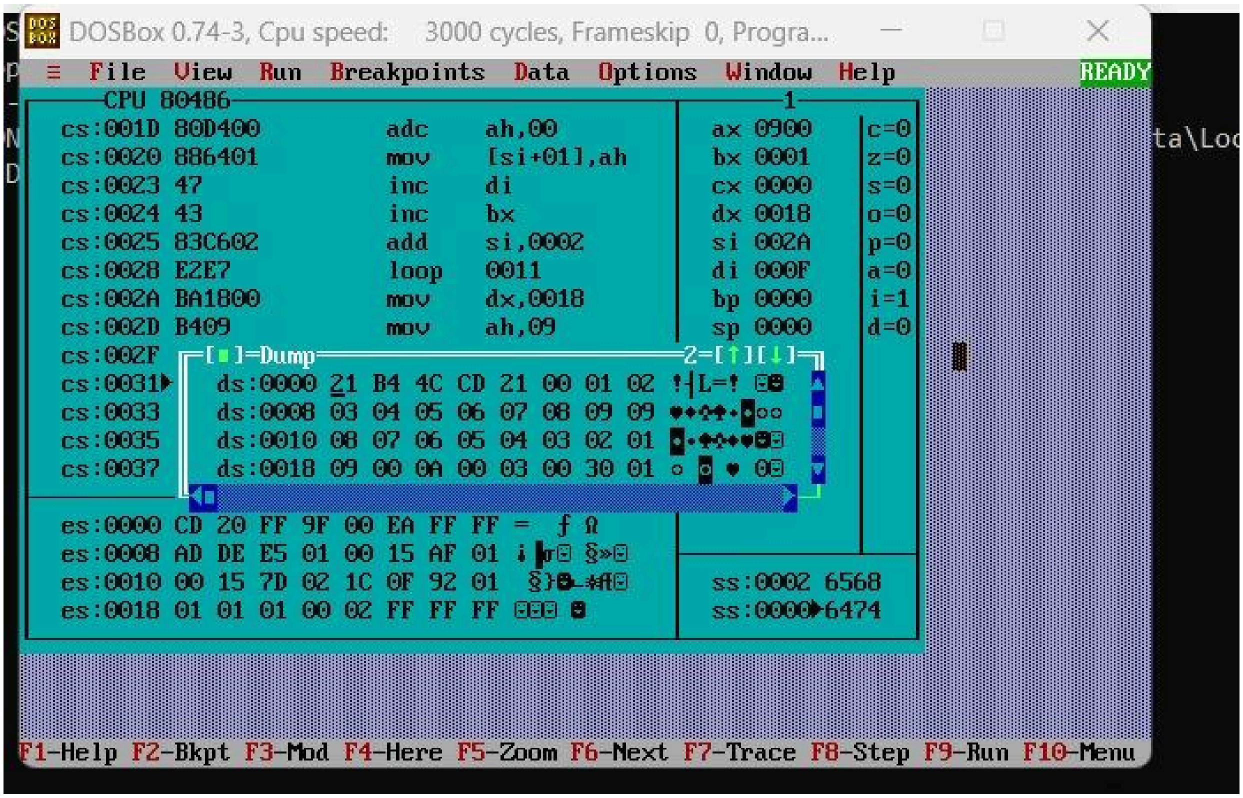
Postlab:

1. **Write a program to Multiply 3 X 3 Matrix.**



```
.model small
.data
matrix1 db 1, 2, 3, 4, 5, 6, 7, 8, 9
matrix2 db 9, 8, 7, 6, 5, 4, 3, 2, 1
result_mult db 9 dup(?)

.code
start:
    mov ax, @data
    mov ds, ax

    mov di, offset matrix1
    mov cx, 9
    mov bx, offset matrix2
    mov si, offset result_mult

multiplication_loop:
    mov al, [di]
    mov ah, 0
    mov bl, [bx]
    mul bl
    add al, [si]
    mov [si], al
    adc ah, 0
    mov [si + 1], ah

    inc di
    inc bx
    add si, 2

    loop multiplication_loop

    mov dx, offset result_mult
    mov ah, 09h
    int 21h

    mov ah, 4Ch
    int 21h

end start
```

Prepared by : Prof. Heenakausar Pendhari


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Progra...
≡ File View Run Breakpoints Data Options Window Help
READY

```
CPU 80486                                                    1
 cs:001D 80D400          adc     ah,00            ax 0900  c=0
 cs:0020 886401          mov     [si+01],ah       bx 0001  z=0
 cs:0023 47              inc     di               cx 0000  s=0
 cs:0024 43              inc     bx               dx 0018  o=0
 cs:0025 83C602          add     si,0002          si 002A  p=0
 cs:0028 E2E7            loop    0011             di 000F  a=0
 cs:002A BA1800          mov     dx,0018          bp 0000  i=1
 cs:002D B409            mov     ah,09            sp 0000  d=0
 cs:002F    [■]=Dump==========================2=[↑][↓]
 cs:0031►      ds:0000 21 B4 4C CD 21 00 01 02 !┤L=!  ☺☻
 cs:0033       ds:0008 03 04 05 06 07 08 09 09 ♥♦♣♠•◘○○
 cs:0035       ds:0010 08 07 06 05 04 03 02 01 ◘•♠♣♦♥☻☺
 cs:0037       ds:0018 09 00 0A 00 03 00 30 01 ○ ◙ ♥ 0☺

 es:0000 CD 20 FF 9F 00 EA FF FF =  ƒ Ω
 es:0008 AD DE E5 01 00 15 AF 01 ¡▌σ☺ §»☺
 es:0010 00 15 7D 02 1C 0F 92 01  §}☻∟☼Æ☺      ss:0002 6568
 es:0018 01 01 01 00 02 FF FF FF ☺☺☺ ☻         ss:0000►6474
```

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