1).Write an assembly language program to perform multiplication of 8-bit data.



**Code:-**

.model small

.stack 100h

.data

num1 db 6h

num2 db 2h

result dw 0

msg db '$'

newline db 0Dh, 0Ah, '$'

.code

main proc

mov ax, @data

mov ds, ax

mov al, num1

mov bl, num2

xor cx, cx

xor dx, dx

multiply:

test bl, 1

jz skip\_add

add cx, ax

skip\_add:

shl ax, 1

shr bl, 1

inc dx

cmp dx, 8

jl multiply

mov result, cx

lea dx, msg

mov ah, 09h

int 21h

mov ax, result

call DisplayResult

mov ah, 4Ch

int 21h

main endp

DisplayResult proc

mov bx, 10

xor cx, cx

convert\_digit:

xor dx, dx

div bx

add dl, '0'

push dx

inc cx

test ax, ax

jnz convert\_digit

display\_digit:

pop dx

mov ah, 02h

int 21h

loop display\_digit

lea dx, newline

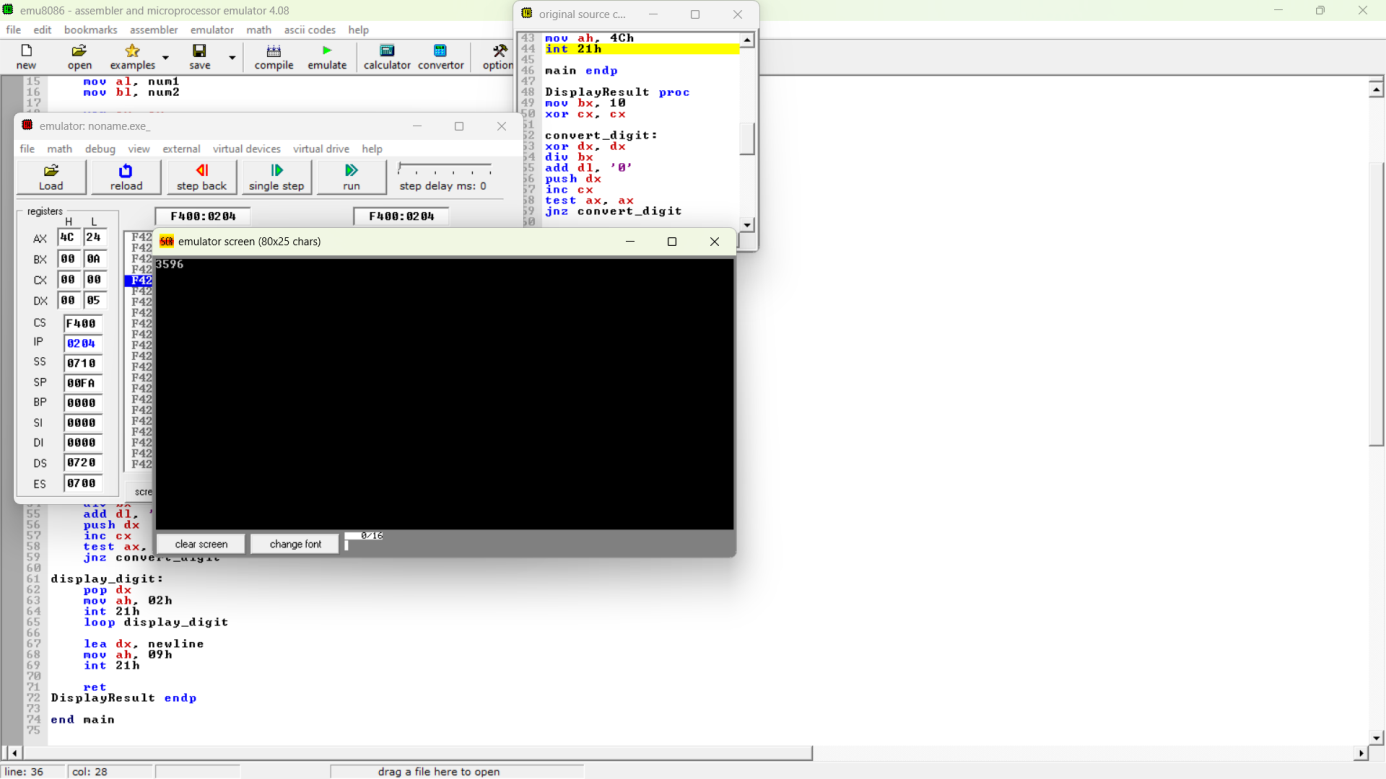
mov ah, 09h

int 21h

ret

DisplayResult endp

end main

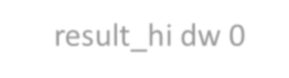
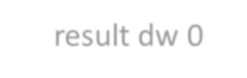
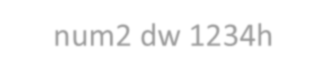
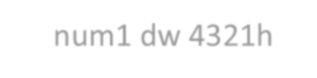
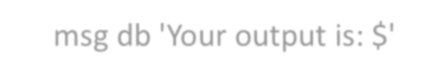
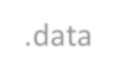
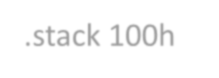
(OUTPUT)  




2. Write a program in assembly language to perform multiplication of 16-bit data.



**Code:-**



.model small

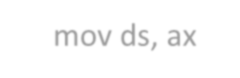
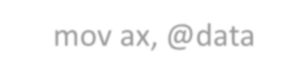
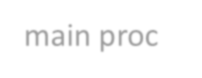
.stack 100h

.data

msg db 'Your output is: $'

num1 dw 4321h num2 dw 1234h result dw 0

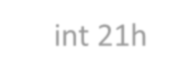
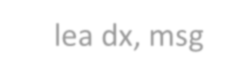
result\_hi dw 0



.code

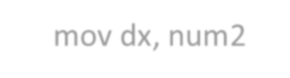
main proc

mov ax, @data mov ds, ax



mov ah, 09h

lea dx, msg int 21h

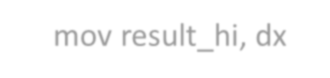


mov ax, num1

mov dx, num2



mul dx

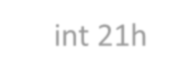


mov result, ax

mov result\_hi, dx



call DisplayHex32

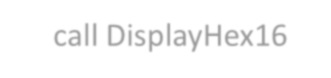
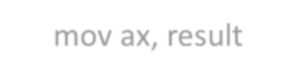
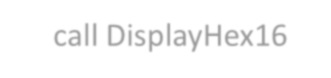
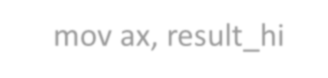


mov ah, 4Ch

int 21h

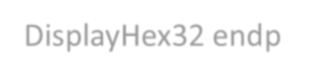


main endp



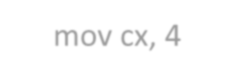
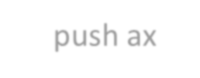
DisplayHex32 proc

mov ax, result\_hi call DisplayHex16 mov ax, result call DisplayHex16



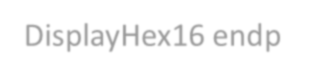
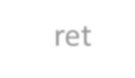
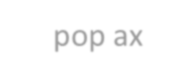
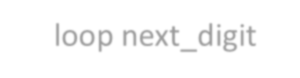
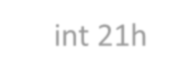
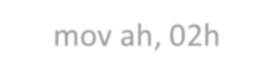
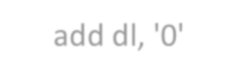
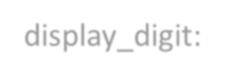
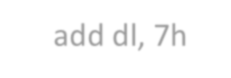
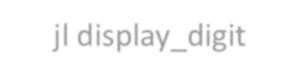
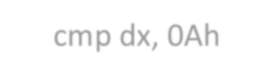
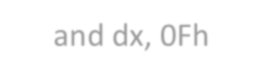
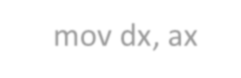
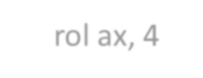
ret

DisplayHex32 endp



DisplayHex16 proc

push ax mov cx, 4



next\_digit:

rol ax, 4

mov dx, ax and dx, 0Fh cmp dx, 0Ah

jl display\_digit add dl, 7h

display\_digit: add dl, '0'

mov ah, 02h int 21h

loop next\_digit pop ax

ret

DisplayHex16 endp



end main



Output:-

