A Lab Project on

**CALCULATOR**

Submitted By:

M.Hamza Farooq (200901019)

Hammad Sahi (200901044)

Submitted to MAM MADIHA



**Problem Statement:**

Construct a basic calculator to perform some arithmetic operations.

**Objective of Our Calculator:**

To perform basic arithmetic operations in less time and efficiently, we need calculator as calculators give us fast and simple calculations. We have made the calculator which perform the arithmetic operations by accepting two inputs from a user as first number and second number with the operators as following:

* Addition (+)
* Subtraction (-)
* Multiplication (\*)
* Division (/)
* Power of a number (number ^ number)

**Block Diagram:**

User have to run the program

Enter first Input

Enter second Input

Press any operand to select from the following operands.

Operation Performed

Answer Displayed

**Source Code**:

include emu8086.inc

org 100h

.data

namee db 0ah,0dh, " .........h&h calculator......... $"

space db 0ah,0dh, "................................................$"

line1 db 0ah,0dh, " take 1st input from the user : $"

line2 db 0ah,0dh, " take 2nd input from the user : $"

line3 db 0ah,0dh, " input the given operator from the following: ^, + , - , \* , / : $"

line4 db 0ah,0dh, " answer = $"

in\_msg db 0ah,0dh, "!!!!! user entered the invalid input !!!!! $"

num1 dw 00h

num2 dw 00h

overflow db 00h

.code

lea dx,space ;to print the string present in space

mov ah,09h ;to display character string

int 21h

lea dx,namee

mov ah,09h

int 21h

lea dx,space

mov ah,09h

int 21h

callc:

mov ax,@data

mov ds,ax

call entering

call msg2

call oprs

mov [si],'&'

call msg5

call answer

entering proc ; Procedure

mov [si],'&' ;source index used as pointer

lea dx,line1

mov ah,09h

int 21h

entering1:

mov ah,01h

int 21h

cmp al,13d ;we are checking wether we entered the 1st input or not

jz msg1 ;if entered,jump on second input

mov ah,al

sub ah,'0'

jc invalid

mov ah,al

mov dh,'9'

sub dh,ah

jc invalid

sub al,'0'

inc si ;incrementing si

mov [si],al

jmp entering1

msg1:

inc si

mov [si],'&'

lea dx,line2

mov ah,09h

int 21h

enter2:

mov ah,01h

int 21h

cmp al,13d

jz exit

mov ah,al

sub ah,'0'

jc invalid

mov ah,al

mov dh,'9'

sub dh,ah

jc invalid

sub al,'0'

inc si

mov [si],al

jmp enter2

exit:

ret

invalid: lea dx,in\_msg ;Display Invalid!!!!!

mov ah,09h

int 21h

hlt

endp

msg2 proc

mov cx,01d

mov bx,00h

msg3:

mov ax,00h

mov al,[si]

mul cx

add bx,ax

mov ax,cx

mov cx,10d

mul cx

mov cx,ax

dec si

cmp [si],'&'

jnz msg3

mov [num2],bx

mov bx,00h

mov dx,00h

dec si

mov cx,01d

msg4:

mov ax,00h

mov al,[si]

mul cx

add bx,ax

mov ax,cx

mov cx,10d

mul cx

mov cx,ax

dec si

cmp [si],'&'

jnz msg4

mov [num1],bx

mov ax,[num1]

mov bx,[num2]

ret

endp

oprs proc

mov cx,ax

lea dx,line3

mov ah,09h

int 21h

mov ah,01h

int 21h

cmp al,'+' ;Now we will compare entered operator with given operators.

jz plus

cmp al,'-'

jz minus

cmp al,'\*'

jz multiply

cmp al,'/'

jz divide

cmp al,'^'

jz power

lea dx,in\_msg

mov ah,09h

int 21h

hlt

plus:

mov ax,cx

mov dx,00h

add ax,bx

adc ax,dx

ret

minus:

mov ax,cx

sub ax,bx

jc ov

jnc nov

ov:neg ax

mov [overflow],01h

ret

nov:ret

multiply:

mov ax,cx

mov dx,00h

mul bx

ret

divide:

mov ax,cx

mov dx,00h

add bx,dx

jz a

div bx

ret

a: print ' error : divide by zero'

jmp callc

power:

mov ax,cx

mov cx,bx

add cx,00h

jz l4

sub cx,01h

jz l1

jnz l2

l1: ret

l2: mov bx,ax

mov dx,00h

l3: mul bx

loop l3

ret

l4: mov ax,01h

ret

endp

msg5 proc

msg6:

mov dx,00h

mov bx,10d

div bx

add dl,'0'

inc si

mov [si],dl

add ax,00h

jnz msg6

endp

answer proc

lea dx,line4

mov ah,09h

int 21h

mov cl,01h

cmp cl,[overflow]

mov [overflow],00h

jz result\_min

jnz result\_print

result\_min: mov dl,'-'

mov ah,02h

int 21h

result\_print:

mov dl,[si]

mov ah,02h

int 21h

dec si

cmp [si],'&'

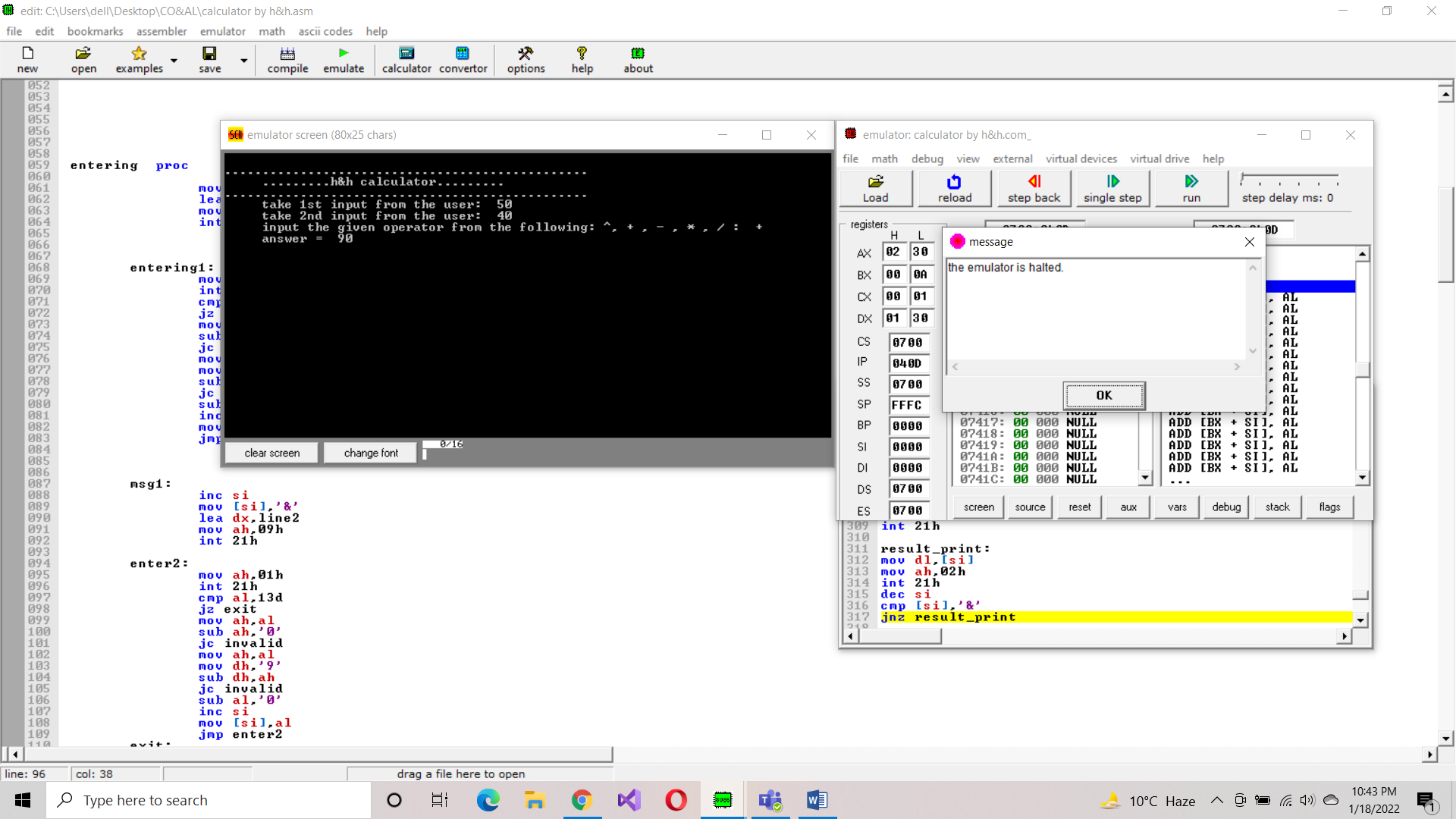
jnz result\_print ;printing result

; we can continue calculato by jmp callc

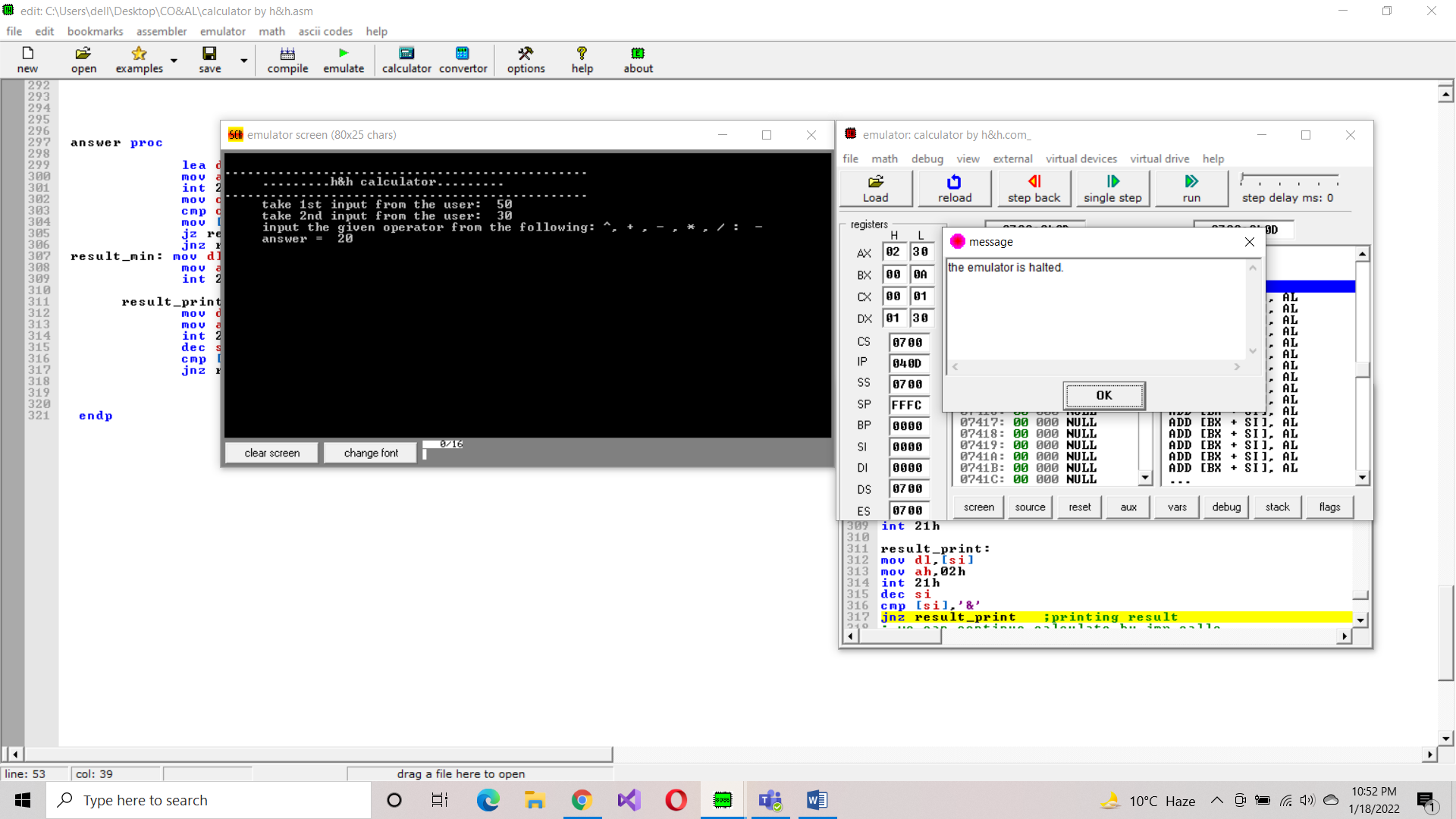
endp

**Testing and Debugging:**

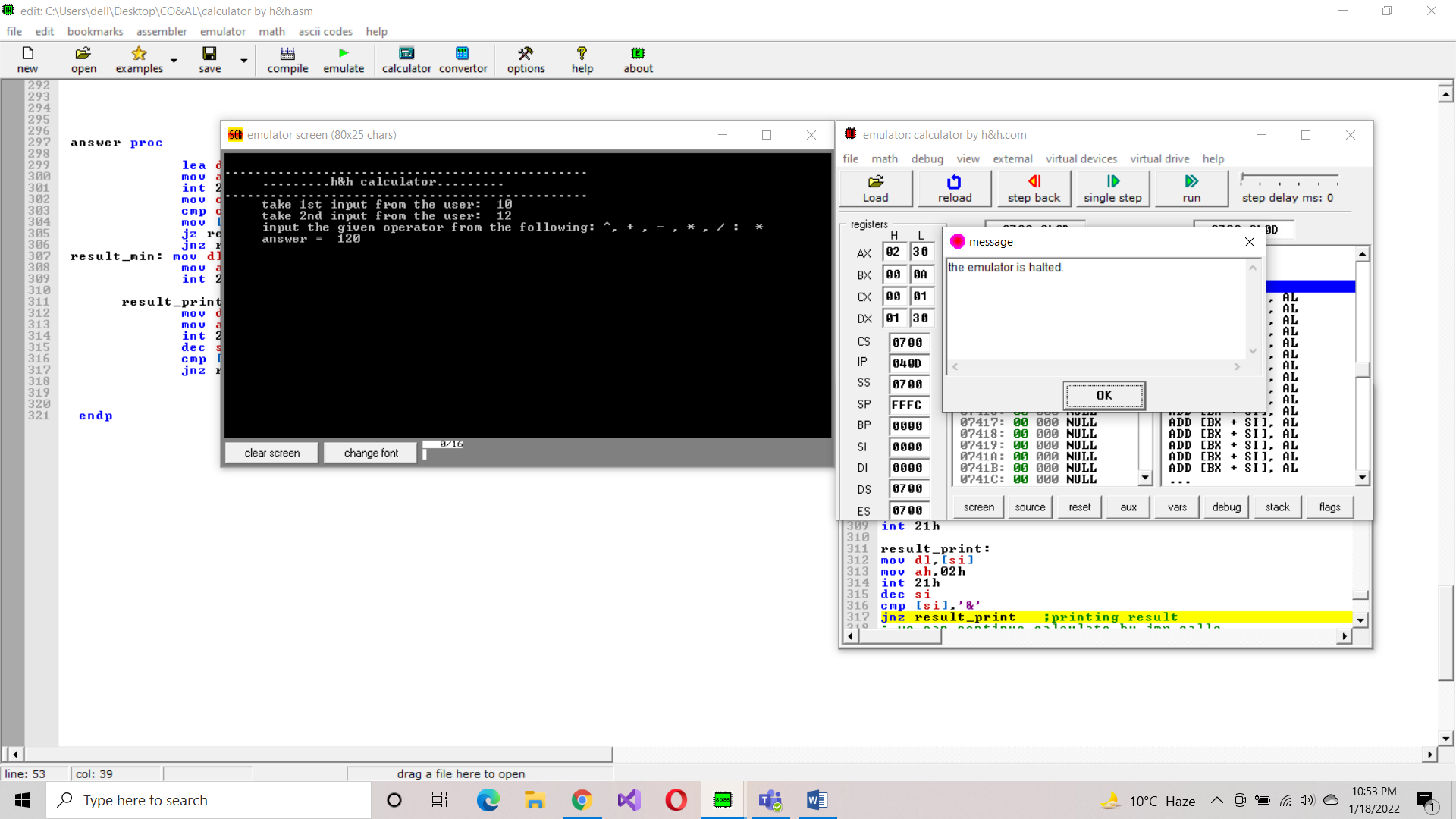
**For Addition:**



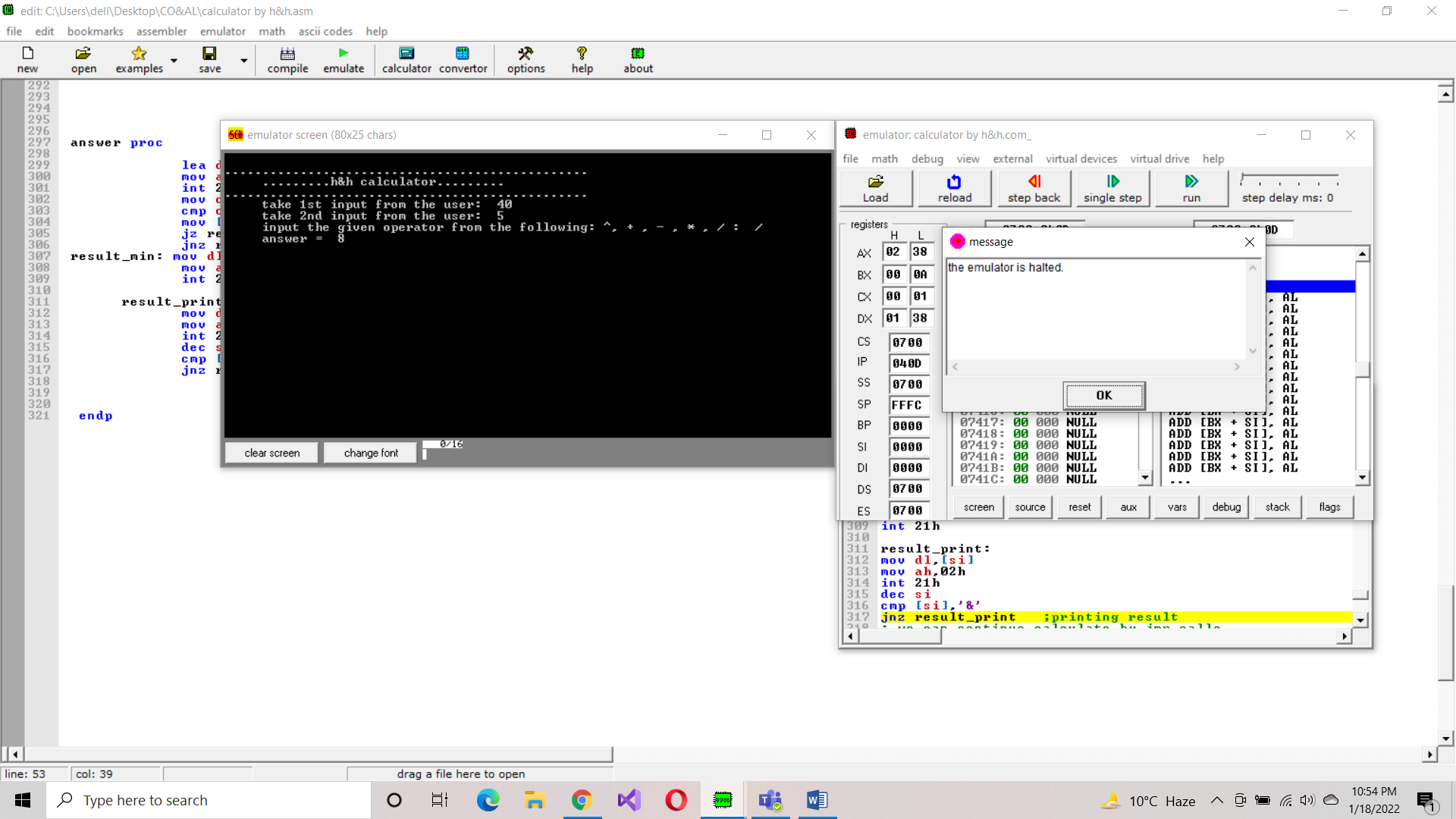
**For Subtraction:**



**For Multiplication:**



**For Division:**



**For Power:**

