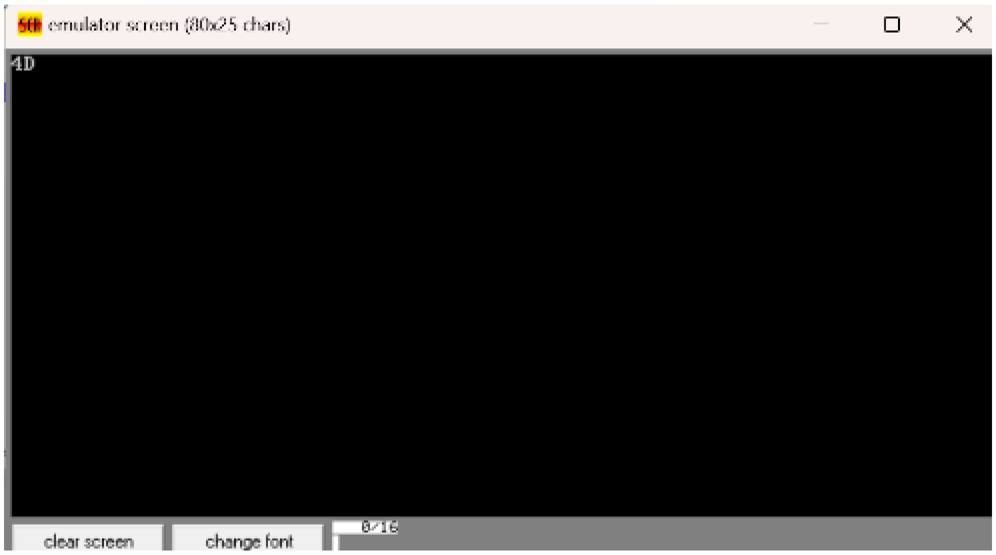
Program to perform addition of 8-bit data.

```
org 100h
num1 db 24h
num2 db 29h

start:
  mov al, num1   ;moving number1 value to AL Register
  add al, num2   ;adding number2 value with existing value in AL register
  mov bl,al

  ;to covert upper nibble(4 bits) of AL to a character
  mov ah,al
  and ah,0F0h  ;mask the lower nibble(all lower bits become 0's)
  shr ah,4     ;shift right by 4 to get upper nibble
  add ah,30h   ;convert to ASCHII digits(0-9)
  cmp ah,39h   ;compare ah value if ah is less than 39h
  jle print_first_digit
  add ah,7     ;covert to ASCHII letter(A-F) if ah>39h

print_first_digit:
  mov dl,ah    ;move first digit to DL for printing
  mov ah,02h   ;BIOS interrupt to display character
  int 21h

  ;to convert lower nibble(4 bits) of AL to a character
  mov ah,bl
  and ah,0Fh   ;mask the upper nibble(all upper nibbles become 0's)
  add ah,30h
  cmp ah,39h
  jle print_second_digit
  add ah,7

print_second_digit:
  mov dl,ah    ;move second digit to DL for printing
  mov ah,02h   ;BIOS interrupt to display character
  int 21h

  ;End the program
  mov ah,4Ch  ;Terminate the program
  int 21h
```

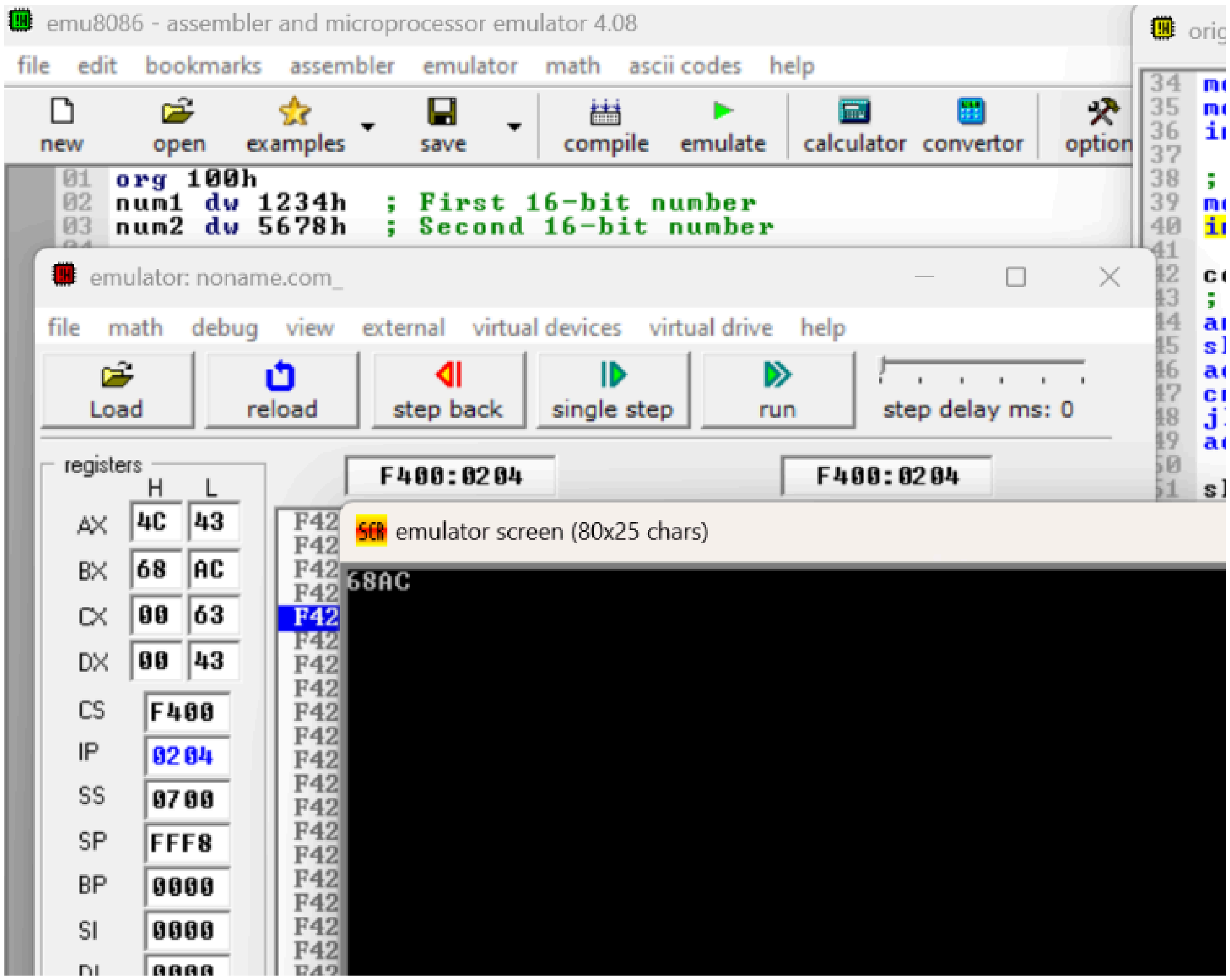


```
4D
```

Program to perform addition of 16-bit data.

```
org 100h
num1 dw 1234h  ; First 16-bit number
num2 dw 5678h  ; Second 16-bit number

start:
   mov ax, num1  ; Load num1 into AX register (16-bit register)
   add ax, num2  ; Add num2 to AX

   ; Store the result in BX so we can convert it to hexadecimal
   mov bx, ax

   ; Convert and print the upper byte (higher 8 bits)
   mov ah, bh   ; Move upper byte of BX to AH
   call convert_to_hex  ; Convert upper nibble to hex
   mov dl, ah   ; Move first character to DL for printing
   mov ah, 02h  ; BIOS interrupt to display character
   int 21h

   mov ah, bh   ; Move upper byte of BX to AH again
   call convert_lower_nibble  ; Convert lower nibble to hex
   mov dl, ah   ; Move second character to DL for printing
   mov ah, 02h  ; BIOS interrupt to display character
   int 21h
```

```
    ; Convert and print the lower byte (lower 8 bits)
    mov ah, bl   ; Move lower byte of BX to AH
    call convert_to_hex  ; Convert upper nibble to hex
    mov dl, ah   ; Move third character to DL for printing
    mov ah, 02h  ; BIOS interrupt to display character
    int 21h

    mov ah, bl   ; Move lower byte of BX to AH again
    call convert_lower_nibble  ; Convert lower nibble to hex
    mov dl, ah   ; Move fourth character to DL for printing
    mov ah, 02h  ; BIOS interrupt to display character
    int 21h

    ; End the program
    mov ah, 4Ch  ; Terminate the program
    int 21h

convert_to_hex:
    ; Mask the upper nibble and convert it to a character
    and ah, 0F0h
    shr ah, 4    ; Shift the upper nibble to the lower nibble
    add ah, 30h  ; Convert to ASCII digit
    cmp ah, 39h  ; Compare if the value is less than or equal to '9'
    jle skip_conversion
    add ah, 7    ; Convert to ASCII letter (A-F)

skip_conversion:
    ret          ; Return from the procedure

convert_lower_nibble:
    ; Mask the lower nibble and convert it to a character
    and ah, 0Fh  ; Mask upper nibble, keep lower nibble
    add ah, 30h  ; Convert to ASCII digit
    cmp ah, 39h  ; Compare if the value is less than or equal to '9'
    jle skip_lower_conversion
    add ah, 7    ; Convert to ASCII letter (A-F)

skip_lower_conversion:
    ret          ; Return from the procedure
```



```
org 100h
num1 dw 1234h   ; First 16-bit number
num2 dw 5678h   ; Second 16-bit number
```

emu8086 - assembler and microprocessor emulator 4.08
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor option
emulator: noname.com_
file math debug view external virtual devices virtual drive help
Load reload step back single step run
step delay ms: 0
emulator screen (80x25 chars)

F400:0204   F400:0204

```
registers
     H   L
AX  4C  43
BX  68  AC
CX  00  63
DX  00  43
CS  F400
IP  0204
SS  0700
SP  FFF8
BP  0000
SI  0000
```

```
68AC
```