



MİKROİŞLEMCİ SİSTEMLERİ

Dr. Öğr. Üyesi Meltem KURT PEHLIVANOĞLU

W-4

8086 Mikroişlemci

Segment ve adres register çiftleri:

CS	IP
SS	SP
	BP
DS	BX
	SI
	DI
ES	DI

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **JMP KOMUTU:**

Koşulsuz dallanma programda istenilen yere atlanır.

JMP operand1

Operand1 burada **etiket** olur. Bu etiket aslında bellek adresidir ve siz etiketin gösterdiği bellek adresine atlamış olursunuz.

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
ORG 100h  
MOV AL, 5
```

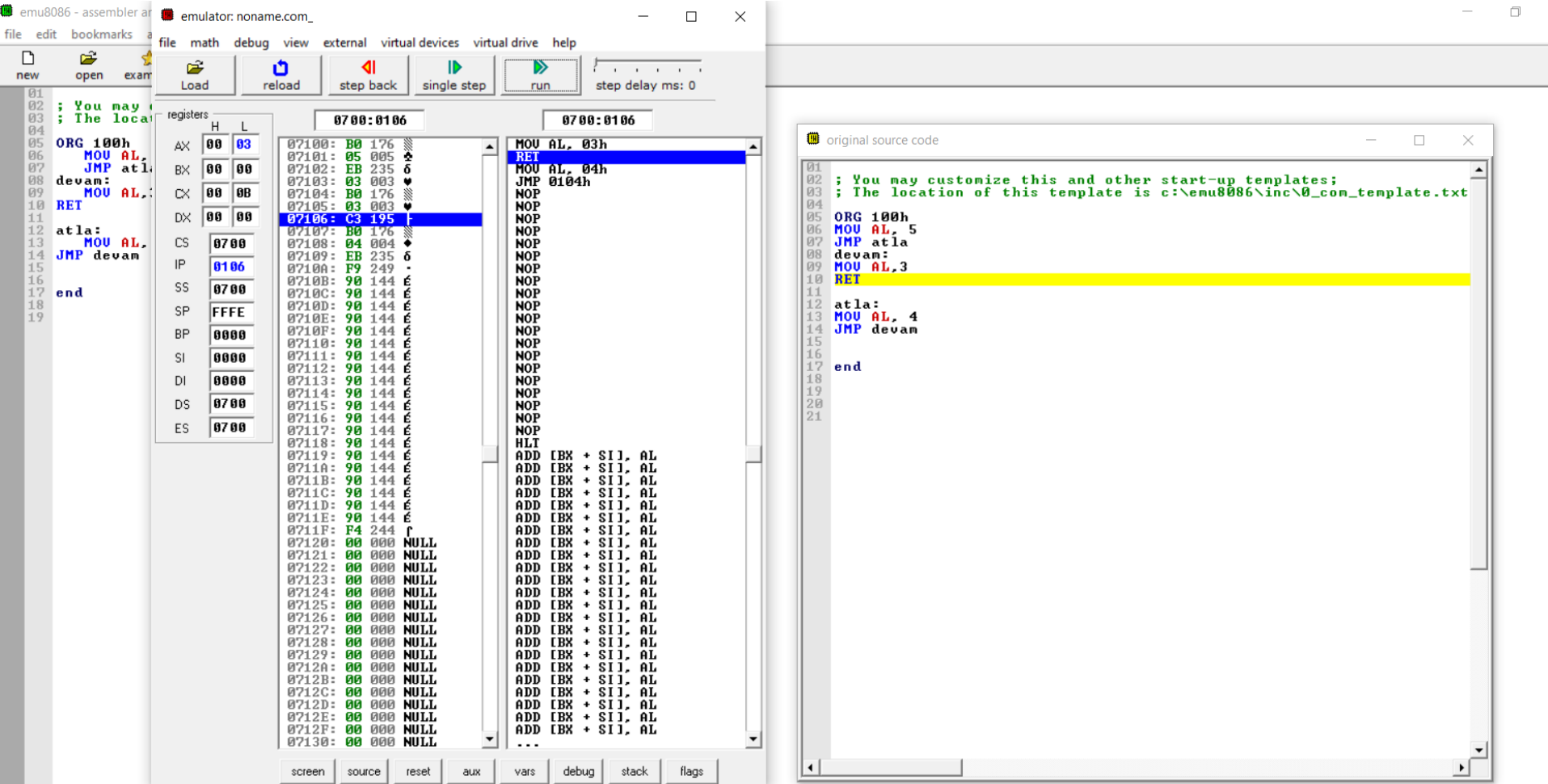
```
JMP atla  
devam:  
MOV AL,3
```

```
RET
```

```
atla:  
MOV AL, 4  
JMP devam
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **LOOP KOMUTU:**

Operand1

Etiket:

- **CX registerına döngünün kaç kez döneceğini atamak zorundasınız.**
- **Komut CX=0 olana kadar devam eder**

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

org 100h

MOV AL, 5

MOV CX, 4 ; dongunun kac kez tekrar edeceğini
belirtiyoruz

dongu1:

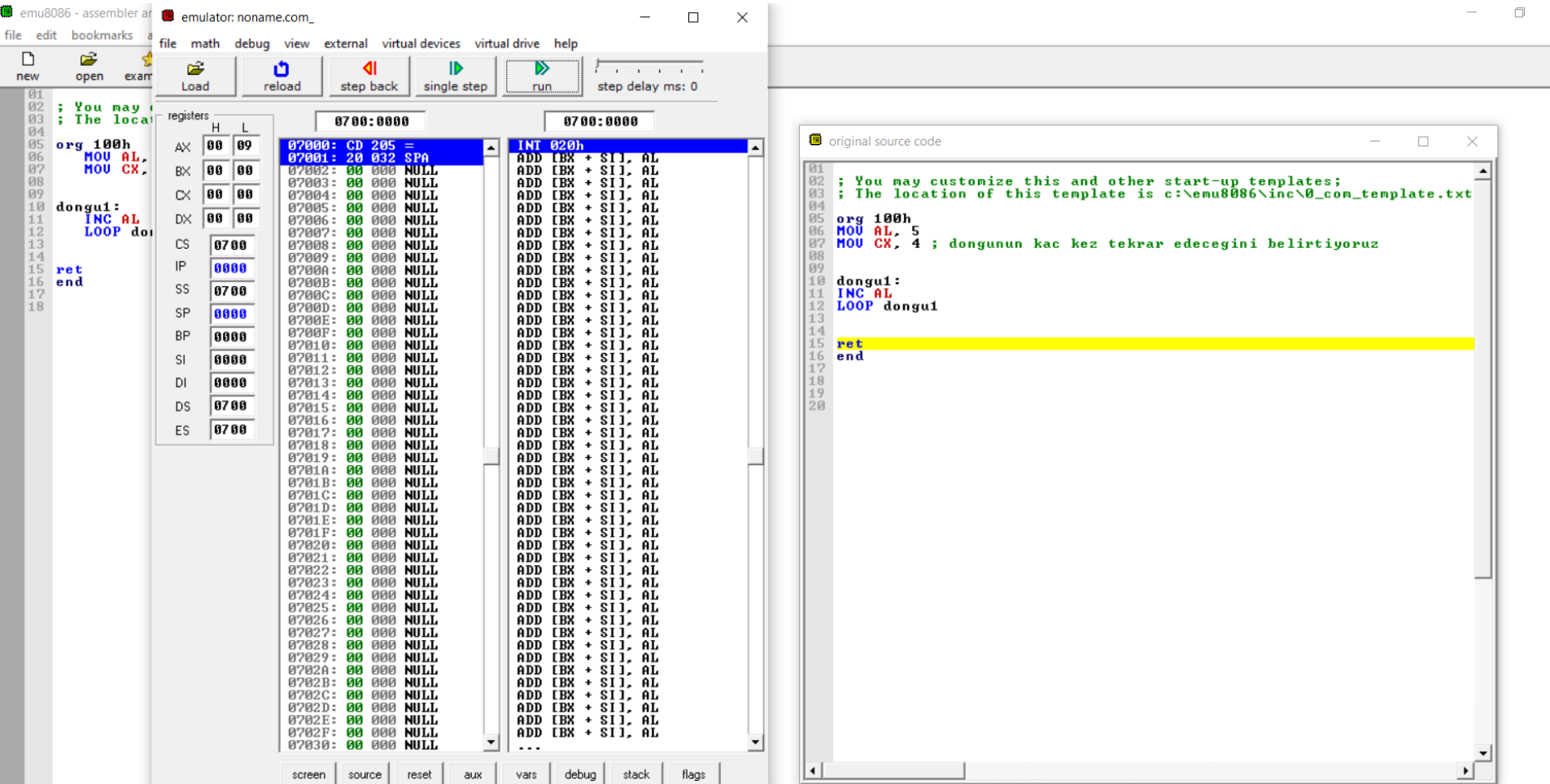
INC AL

LOOP dongu1

ret

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

ARİTMETİKSEL KOMUTLAR:

- **ADD** operand1,operand2

$\text{operand1} = \text{operand1} + \text{operand2}$

- **ADC** operand1,operand2

$\text{operand1} = \text{operand1} + \text{operand2} + \text{CF}$ (carry flag)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h
```

```
mov al,10  
add al,20 ; al=1E
```

```
mov al,255 ; maksimum alınacak deger  
add al,1 ; IF=1 zaten varsayılan geliyor  
; CF=1  $255+1=256$  isaretsiz sayılarda tasma oldu  
; ZF=1 oldu  $256-256=0$  islem sonucu 0 oldugundan ZF aktif oldu  
; PF=1 1 bitlerin sayısı ciftse aktifti, 8 bitte 0 tane 1 var cift olarak goruyor PF aktif oluyor  
; AF=1 sondaki 4 bitte tasma oldugundan AF aktif olur  $1111\ 1111 + 0001$  toplaması
```

```
mov al,255  
add al,5 ;  $255+5=260-256=4$  AL de 04 bulunur
```

```
mov al,-2  
add al,255 ;  $255-2=253$  ; AL de FD
```

```
mov ax,258  
add ax,5 ;  $258+5=263$  ; AX 01 07
```

```
add sayi1,3 ; sayi1=253  
mov bl,sayi1 ; BL=FD 253 HEX karsiligi
```

```
add [sayi1],5 ;  $253+5=258-256=2$   
mov bh,sayi1 ; BH=02 olur
```

```
ret
```

```
sayi1 db 250
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **SUB** operand1,operand2

operand1=operand1-operand2

- **SBB** operand1,operand2

operand1=operand1-operand2-CF

```
org 100h
```

```
mov al,1
```

```
sub al,3 ; -2 256-2=254 FE olur
```

```
ret
```

```
sayi1 db 250
```

```
org 100h
```

```
mov al,1
```

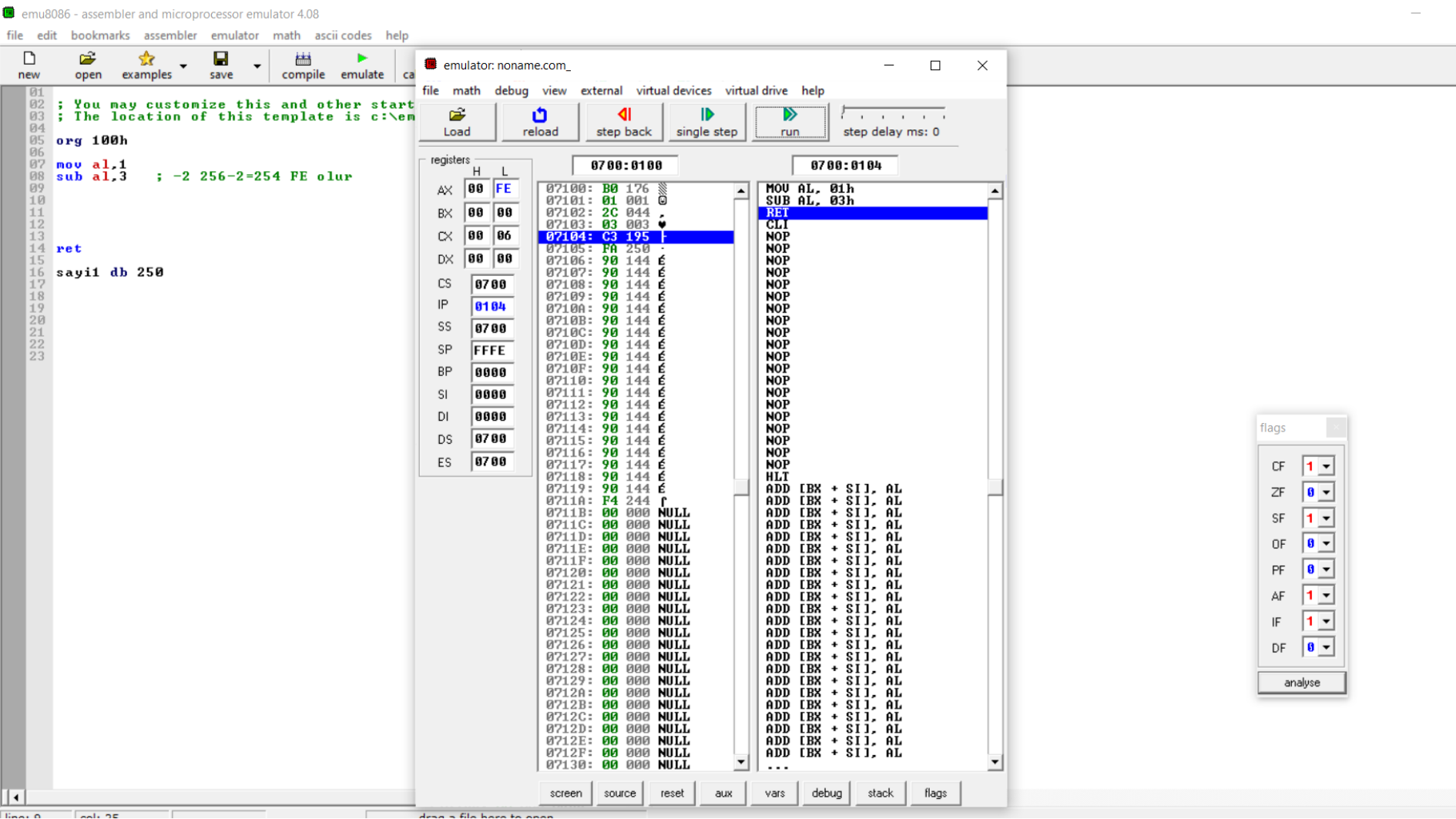
```
sub al,3 ; -2 256-2=254 FE olur  
; CF=1
```

```
sbb al,1 ; FE-1-1= FC
```

```
ret
```

```
sayi1 db 250
```

EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

emu8086 - assembler and microprocessor emulator 4.08

The screenshot displays the EMU 8086 Microprocessor Emulator interface. The main window is titled "emulator: noname.com_". It features a menu bar with options: file, math, debug, view, external, virtual devices, virtual drive, and help. Below the menu bar is a toolbar with buttons for Load, reload, step back, single step, run, and a step delay slider set to 0 ms.

The interface is divided into several panes:

- Registers:** A table showing the current values of the 16-bit registers. The AX register is highlighted with a value of 00 FC. The IP (Instruction Pointer) is 0106.
- Memory:** A table showing the contents of memory locations starting from 07100h. The location 07106h is highlighted, containing the value C3 195.
- Source Code:** A window titled "original source code" showing the assembly code being executed. The code includes comments and instructions like `mov al,1`, `sub al,3`, `sbb al,1`, and `ret`.
- Flags:** A small window titled "flags" showing the status of various flags: CF (0), ZF (0), SF (1), OF (0), PF (1), AF (0), IF (1), and DF (0). An "analyse" button is at the bottom.

The assembly code in the source code window is as follows:

```
01 ; You may customize this and other start-up templates;  
02 ; The location of this template is c:\emu8086\inc\0_com_template.txt  
03 org 100h  
04  
05  
06 mov al,1  
07 sub al,3 ; -2 256-2=254 FE olur  
08 ; CF=1  
09 sbb al,1 ; FE-1-1= FC  
10  
11 ret  
12  
13 sayi1 db 250  
14  
15  
16  
17  
18  
19  
20  
21
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **CBW (Convert byte into word):** 8-bitlik değeri 16-bitlik değere genişletir

Operand almaz, AL nin yüksek değerli 8. bitini, AH içine yayar

- **CWD (Convert Word to Double word):** 16-bitlik değeri 32-bitlik değere genişletir

Operand almaz, AX içindeki yüksek değerli 16. biti DX içine yayar

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

org 100h

mov al,-3 ;FD=11111101 8. biti 1 AH icindeki 8 biti de 1 yapar (1111 1111) o yuzden
AH FF olur
cbw

mov ax,0
mov ax,0FF5Fh ;FF5F: 11111111 01011111 16. bit 1 oldugu icin
cwd ;DX icini 1 ile doldurur DX=FF FF (1111 1111 1111 1111) olur

ret
sayi1 db 5

ALIŞTIRMA SORULARI

ALIŞTIRMA SORULARI

sayilar=2,4,6,3 dizisindeki her bir elemanı 1 arttırarak 'sayilar2' dizisine tersten aktaran kodu yazınız?

sayilar2 = 4,7,5,3

ÇÖZÜM

org 100h

MOV CX,4

MOV SI,0

MOV DI,3

dongu:

MOV AL,0

MOV BL,0

MOV AL, sayilar+SI

MOV BL,AL

INC BL

MOV sayilar2+DI,BL

INC SI

DEC DI

LOOP dongu

ret

sayilar db 2,4,6,3

sayilar2 db 4 dup(?)

ÇÖZÜM 2

```
org 100h
```

```
LEA SI, sayilar  
LEA DI, sayilar2
```

```
MOV CX,4  
MOV BP, 3
```

```
dongu:  
MOV AL, 0  
MOV AL, [SI]  
INC AL  
MOV [DI+BP],AL  
INC SI  
DEC BP  
LOOP dongu
```

```
ret
```

```
sayilar db 2,4,6,3
```

```
sayilar2 db 4 dup(?)
```