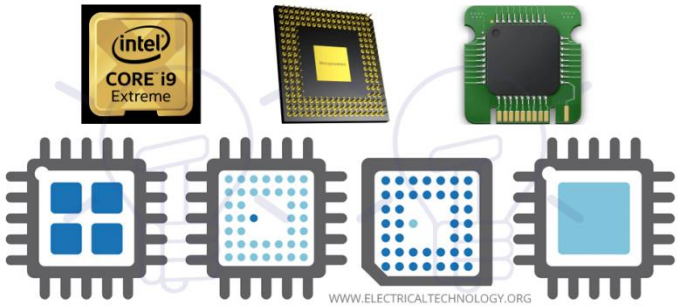


MİKROİŞLEMCİLER

Dr. Meltem KURT PEHLİVANOĞLU

W-13

MİKROİŞLEMCİLER

Digital Logic +

Digital Design +

Computer Architecture +

Microprocessors +

Microcontrollers +

Assembly Language Programming(8086)

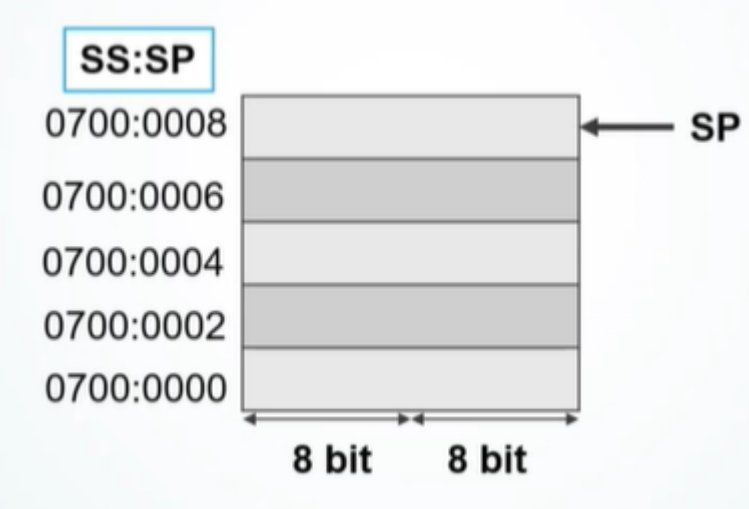
8086 16-Bit Mikroişlemci

- Segment ve adres register çiftleri:
- CS:IP
- SS:SP SS:BP
- DS:BX DS:SI
- DS:DI
- ES:DI

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- Stack (yığın): Geçici verileri tutmak için kullanılan bellek alanıdır.
- LIFO (Last In First Out) mantığı ile çalışır. Yani son giren ilk çıkar
- Normalde her RAM hücresi 8 bit (1 byte) yer kaplıyor ancak Stack içinde her eleman 16 bit (2 byte) olarak tutuluyor. Diğer bir ifadeyle ardışık 8 bitlik 2 RAM hücresi işgal eder.



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- CF(carry flag): Elde varsa 1 olur.
- ZF(zero flag): Herhangi bir işlem sonucunda 0 elde ediliyorsa ZF 1 olur
- SF(sign flag): ALU tarafından gerçekleştirilen bir işlemin sonucu eğer negatif çıkıyorsa SF 1 olur
- OF(overflow flag): İşaretli sayılarda işlem sonucu işaretli sayı aralığını aşıyorsa taşma bayrağı 1 olur (8-bitlik işaretli sayılar için en küçük değer -128, en büyük değer +127)
- PF(parity flag): İşlem sonucunda bulunan '1' bitlerinin sayısı çift ise PF 1 olur. **Sonuç 16-bit olsa bile düşük değerlikli 8-bit ele alınır.**
- AF(auxiliary flag): İşaretsiz sayılarda yapılan işlemlerdeki düşük değerlikli 4 bitte taşma meydana gelirse AF 1 olur.
- DF(direction flag): Diziler gibi ardışık verilerde özellikle string işlemlerinde kullanılan komutların ileri yönlü mü yoksa geri yönlü mü çalışacağını belirlemek için kullanılır. DF=0 iken ileri yönlü (düşük adresten yüksek adrese) işlem yapılır, DF=1 iken geri yönlü (yüksek adresten düşük adrese). Varsayılan 0 değeridir.
- IF (interrupt flag): Varsayılan olarak aktif bu sayede kesmelere izin veriyor. Örneğin klavyeden değer okuma, ekrana metin yazdırma vb.

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

DS VE ES KULLANIMI

- COM dosyalarında tek bir bölüt vardır ve bu bölüt 64KB ile sınırlıdır
- Segment registerları başlangıçta aynı adresi işaret ederler

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

SREG: OFFSET

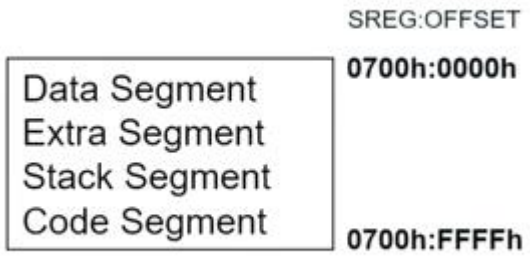
DS : SI

ES : DI

SS : SP

CS : IP

```
org 100h

LEA SI, sayi1
LEA DI,sayi2

MOV AL,[DI]
MOV BL,DS:[DI]
MOV CL,ES:[DI]

ret

sayi1 db 5
sayi2 db 4
sayi3 db 3
```



SREG: OFFSET

DS : SI

ES : DI

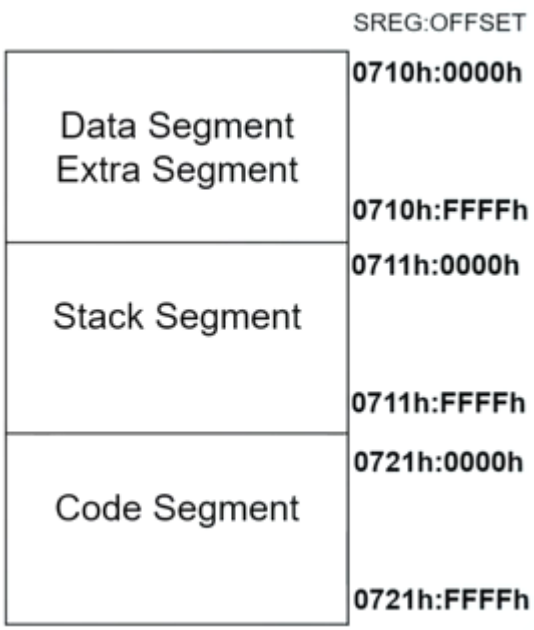
SS : SP

CS : IP

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- Exe dosyasında tüm segmentler ayrılmıştır.
- Her segment 64 KB yer kaplar




```
; multi-segment executable file template.

data segment
   ; veri tanimlamalar data segmentte yapilir

   sayi1 db 5
   sayi2 db 4
   sayi3 db 3

ends



stack segment
   dw   128  dup(0)
ends



code segment

start:
; set segment registers:
   mov ax, data
   mov ds, ax     ; DS ve ES ayni yeri gosterecek
   mov es, ax



   LEA SI, sayi1
   LEA DI,sayi2

   MOV AL,[DI]
   MOV BL,DS:[DI]
   MOV CL,ES:[DI]


   mov ax, 4c00h ; exit to operating system.
   int 21h      ; exit to operating system.
ends

end start ; set entry point and stop the assembler.
```

SREG: OFFSET

DS : SI

ES : DI

SS : SP

CS : IP

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
data segment

   sayi1 db 5
   sayi2 db 4
   sayi3 db 3

ends


extra segment
   sayi4 db 6
   sayi5 db 7

ends


stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax


   LEA SI,sayi1
   LEA DI,sayi2

   MOV AL,[DI]
   MOV CL,ES:[DI]


 mov ax, 4c00h ; exit to operating system.
 int 21h      ; exit to operating system.
ends

end start ; set entry point and stop the assembler.
```

SREG: OFFSET
DS : SI
ES : DI
SS : SP
CS : IP

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **STRING (DİZİ) KOMUTLARI**

MOVSB: Operand almaz

DS:[SI] da bulunan byte olarak tanımlı veriyi ES:[DI] ile gösterilen yere kopyalar.

DS kaynak, ES hedef olur.

DF(direction flag) bayrağına göre SI ve DI güncellenir.
(DF=0 SI=SI+1 ve DI=DI+1, DF=1 SI=SI-1 ve DI=DI-1)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
data segment     ;DS:SI

   kaynak db 1,2,3,4


ends


extra segment   ; ES:DI

   hedef db 4 dup(0)

ends


stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax


   cld   ; df=0
   mov cx,4
   lea SI,kaynak
   lea DI,hedef

   dongu:
   movsb  ; SI ve DI otomatik olarak artar
   loop dongu



 mov ax, 4c00h ; exit to operating system.
 int 21h       ; exit to operating system.
ends

end start ; set entry point and stop the assembler.
```

SREG: OFFSET
DS : SI
ES : DI
SS : SP
CS : IP

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

MOVSW: Operand almaz

DS:[SI] da bulunan word olarak tanımlı veriyi ES:[DI] ile gösterilen yere kopyalar.

DS kaynak, ES hedef olur.

DF(direction flag) bayrağına göre SI ve DI güncellenir.
(DF=0 SI=SI+2 ve DI=DI+2, DF=1 SI=SI-2 ve DI=DI-2)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

LODSB: Operand almaz

DS:[SI] da bulunan byte olarak tanımlı veriyi AL registerına kopyalar.

DS kaynak, AL hedef olur.

DF(direction flag) bayrağına göre SI güncellenir.

(DF=0 SI=SI+1, DF=1 SI=SI-1)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
data segment     ;DS:SI

   kaynak db 1,2,3,4


ends


extra segment   ; ES:DI



ends


stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax


   cld   ; df=0
   mov cx,4
   lea SI,kaynak

   dongu:
   lodsb  ; SI otomatik olarak artar
   loop dongu



 mov ax, 4c00h ; exit to operating system.
 int 21h      ; exit to operating system.
ends

end start ; set entry point and stop the assembler.
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

LODSW: Operand almaz

DS:[SI] da bulunan word olarak tanımlı veriyi AX registerına kopyalar.

DS kaynak, AX hedef olur.

DF(direction flag) bayrağına göre SI güncellenir.

(DF=0 SI=SI+2, DF=1 SI=SI-2)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

STOSB: Operand almaz

AL içinde bulunan byte olarak tanımlı veriyi ES:[DI] ile gösterilen yere kopyalar.

AL kaynak, ES:[DI] hedef olur.

DF(direction flag) bayrağına göre DI güncellenir.

(DF=0 DI=DI+1, DF=1 DI=DI-1)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
data segment     ;DS:SI

ends


extra segment   ; ES:DI

    hedef db 4 dup(0)

ends


stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax


   cld   ; df=0
   mov cx,4
   lea DI,hedef

   dongu:
   mov al,cl
   stosb  ; DI otomatik olarak artar
   loop dongu



 mov ax, 4c00h ; exit to operating system.
 int 21h      ; exit to operating system.
ends

end start ; set entry point and stop the assembler.
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

STOSW: Operand almaz

AX içinde bulunan word olarak tanımlı veriyi ES:[DI] ile gösterilen yere kopyalar.

AX kaynak, ES:[DI] hedef olur.

DF(direction flag) bayrağına göre DI güncellenir.

(DF=0 DI=DI+2, DF=1 DI=DI-2)

SREG: OFFSET
DS : SI
ES : DI
SS : SP
CS : IP

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

CMPSB: Operand almaz

DS:[SI] ile gösterilen yerde bulunan byte olarak tanımlı veriyi ES:[DI] ile gösterilen yerdeki veriyle karşılaştırır.

Temelde çıkarma işlemi yapar

DF(direction flag) bayrağına göre SI ve DI güncellenir.

(DF=0 SI=SI+1 ve DI=DI+1, DF=1 SI=SI-1 ve DI=DI-1)

SREG: OFFSET
DS : SI
ES : DI
SS : SP
CS : IP

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

CMPSW: Operand almaz

DS:[SI] ile gösterilen yerde bulunan word olarak tanımlı veriyi ES:[DI] ile gösterilen yerdeki veriyle karşılaştırır.

Temelde çıkarma işlemi yapar

DF(direction flag) bayrağına göre SI ve DI güncellenir.

(DF=0 SI=SI+2 ve DI=DI+2, DF=1 SI=SI-2 ve DI=DI-2)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

SCASB: Operand almaz

AL içinde bulunan byte olarak tanımlı veriyi ES:[DI] ile gösterilen yerdeki veriyle karşılaştırır.

Temelde çıkarma işlemi yapar

DF(direction flag) bayrağına göre DI güncellenir.

(DF=0 DI=DI+1, DF=1 DI=DI-1)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

SCASW: Operand almaz

AX içinde bulunan word olarak tanımlı veriyi ES:[DI] ile gösterilen yerdeki veriyle karşılaştırır.

Temelde çıkarma işlemi yapar

DF(direction flag) bayrağına göre DI güncellenir.

(DF=0 DI=DI+2, DF=1 DI=DI-2)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

TEKRARLAMA KOMUTLARI

- REP operand1 (operand1: MOVSB,MOVSW,LODSB,LODSW,STOSB,STOSWkomutlarının CX=0 olana kadar tekrarlar)

- REPE operand1 (operand1: CMPSB, CMPSW, SCASB, SCASW komutlarının ZF=1 olduğu sürece ve CX=0 olana kadar tekrarlar)

- REPZ operand1 (operand1: CMPSB, CMPSW, SCASB, SCASW komutlarının ZF=1 olduğu sürece ve CX=0 olana kadar tekrarlar)

(REPE-REPZ Kısaca eşitlik olduğu sürece tekrar etmesi gereken durumlarda)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

TEKRARLAMA KOMUTLARI

- REPNE operand1 (operand1: CMPSB, CMPSW, SCASB, SCASW komutlarının ZF=0 olduğu sürece ve CX=0 olana kadar tekrarlar)

- REPNZ operand1 (operand1: CMPSB, CMPSW, SCASB, SCASW komutlarının ZF=0 olduğu sürece ve CX=0 olana kadar tekrarlar)

(REPNE-REPNZ kısaca eşitlik olmadığı sürece tekrar etmesi gereken durumlarda)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
data segment     ;DS:SI

   kaynak db 1,2,3,4


ends


extra segment   ; ES:DI

   hedef db 4 dup(0)

ends


stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax


   cld   ; df=0
   mov cx,4
   rep movsb



 mov ax, 4c00h ; exit to operating system.
 int 21h      ; exit to operating system.
ends

end start ; set entry point and stop the assembler.
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU 1: metin db ‘mikroislemciler dersi’ metnini boşluk karakterine göre ikiye ayıran birinci parçayı ‘metin1’ değişkenine ikinci parçayı ‘metin2’ değişkenine atayan 8086 Assembly kodunu yazınız

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
data segment     ;DS:SI

   metin db 'mikroislemciler dersi'
   metin1 db 15 dup(0)
   metin2 db 15 dup(0)

ends



stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov es, ax



   cld  ; df=0
   mov cx,21
   lea SI,metin
   lea DI,metin1
   mov bl, 20h ; bl de bosluk karakterinin hex degeri var

   dongu:

   movsb    ; DS:[SI] icindeki veriyi ES:[DI] icine aktarir
   cmp [SI],bl
   JE ikinci
   DEC CX
   jmp dongu

   ikinci:
   DEC CX ; JE ikinci ye atladiktan sonra CX in azalmasi gerekir
   DEC CX ; boslugu yazmadigimiz icin CX azaliyor
   inc SI ; bosluktan sonraki karakteri aliyoruz
   lea DI,metin2
   rep movsb
   jmp bitir



 bitir:
 mov ax, 4c00h ; exit to operating system.
 int 21h      ; exit to operating system.
ends

end start ; set entry point and stop the assembler.
```

```
data segment     ;DS:SI

   metin db 'mikroislemciler dersi'


ends


extra segment
   metin1 db 20 dup(0)
   metin2 db 20 dup(0)
ends



stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax



   cld  ; df=0
   mov cx,21
   lea SI,metin
   lea DI,metin1
   mov bl, 20h ; bl de bosluk karakterinin hex degeri var

   dongu:

   movsb    ; DS:[SI] icindeki veriyi ES:[DI] icine aktarir
   cmp [SI],bl
   JE ikinci
   DEC CX
   jmp dongu

   ikinci:
   DEC CX ; JE ikinci ye atladiktan sonra CX in azalmasi gerekir
   DEC CX ; boslugu yazmadigimiz icin CX azaliyor
   inc SI ; bosluktan sonraki karakteri aliyoruz
   lea DI,metin2
   rep movsb
   jmp bitir



 bitir:
 mov ax, 4c00h ; exit to operating system.
 int 21h      ; exit to operating system.
ends

end start ; set entry point and stop the assembler.
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU 2: sayi db 2,3,3,9 dizisinde AL registerı içinde bulunan değeri (AL=3) arayan ve bu degerin dizi içinde kaç kez geçtiğini ‘ara’ değişkenine yazan programı dizi komutları kullanarak 8086 Assembly kodunu yazınız

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
data segment     ;DS:SI
    ara db 0
ends


extra segment   ; ES:[DI]
       sayi db 2,3,3,9
ends


stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax


   lea DI,sayi
   mov al,3  ; 3 sayisini arayalim
   mov cx,4

   dongu:
   JCXZ bitir
   SCASb   ; AL ile ES:[DI] karsilastiriyor
   JE ayni
   DEC CX
   JMP dongu

   ayni:
   DEC CX
   inc ara
   jmp dongu


 bitir:
 mov ax, 4c00h ; exit to operating system.
 int 21h       ; exit to operating system.



ends

end start ; set entry point and stop the assembler.
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **PROSEDÜRLER VE MAKROLAR**

```
org 100h
mov ax,2

CALL ProsedurAdi
CALL ProsedurAdi

ret

ProsedurAdi PROC
    add ax,2
    mov bx,ax
    RET
ProsedurAdi ENDP

END
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU 3: metin db ‘bu dizi icindeki bosluklari bulun’ dizisindeki boşluk karakterlerinin sayisini boslukbul fonksiyonu ile bulup bosluksayisi değişkenine bu sayıyı atan 8086 Assembly kodunu yazınız

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
; fonksiyonsuz

data segment     ;DS:SI
    bosluksayisi db 0
ends


extra segment   ; ES:[DI]
      metin db 'bu dizi icindeki bosluklari bulun'
ends


stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax


   lea DI,metin
   mov al,20h  ; bosluk karakteri hex degeri   mov al," " yazabilirsiniz
   mov cx,33



   dongu:
   JCXZ bitir
   SCASb   ; AL ile ES:[DI] karsilastiriyor
   JE bosluk
   DEC CX
   JMP dongu

   bosluk:
   DEC CX
   inc bosluksayisi
   jmp dongu


 bitir:
 mov ax, 4c00h ; exit to operating system.
 int 21h      ; exit to operating system.



ends

end start ; set entry point and stop the assembler.
```

```
data segment     ;DS:SI
    bosluksayisi db 0
ends


extra segment   ; ES:[DI]
      metin db 'bu dizi icindeki bosluklari bulun'

ends


stack segment
   dw   128  dup(0)
ends


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax


   lea DI,metin
   mov al,20h  ; bosluk karakteri hex degeri   mov al," " yazabilirsiniz
   mov cx,33


   call boslukbul



 bitir:
 mov ax, 4c00h ; exit to operating system.
 int 21h      ; exit to operating system.


boslukbul proc

  dongu:
   JCXZ geri
   SCASb   ; AL ile ES:[DI] karsilastiriyor
   JE bosluk
   DEC CX
   JMP dongu

   bosluk:
   DEC CX
   inc bosluksayisi
   jmp dongu


   geri:
   ret

boslukbul endp



ends

end start ; set entry point and stop the assembler.
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU 4: metin1 db ‘bir dizi’ ve metin2 db ‘bir Dizi’ dizileri ‘karsilastir’ fonksiyonuyla karşılaştırıp aynı elemanların sayısını ‘ayni’ değişkeninde tutan 8086 Assembly kodunu yazınız

8086 16-Bit Mikroişlemci

EMU 8086-
MICROPROCESSOR EMULATOR

```
data segment     ;DS:SI
    metin1 db 'bir dizi'
    ayni db 0
ends

extra segment   ; ES:[DI]
      metin2 db 'bir Dizi'

ends

stack segment
   dw   128  dup(0)
ends

code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax

   lea SI,metin1
   lea DI,metin2
   mov cx,8

   call karsilastir

 bitir:
 mov ax, 4c00h ; exit to operating system.
 int 21h       ; exit to operating system.

karsilastir proc

  dongu:
   JCXZ geri
   CMPSB   ; DS:[SI] ile ES:[DI] karsilastiriyor
   JE aynibul
   DEC CX
   JMP dongu

   aynibul:
   DEC CX
   inc ayni
   jmp dongu

   geri:
   ret

karsilastir endp

ends

end start ; set entry point and stop the assembler.
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

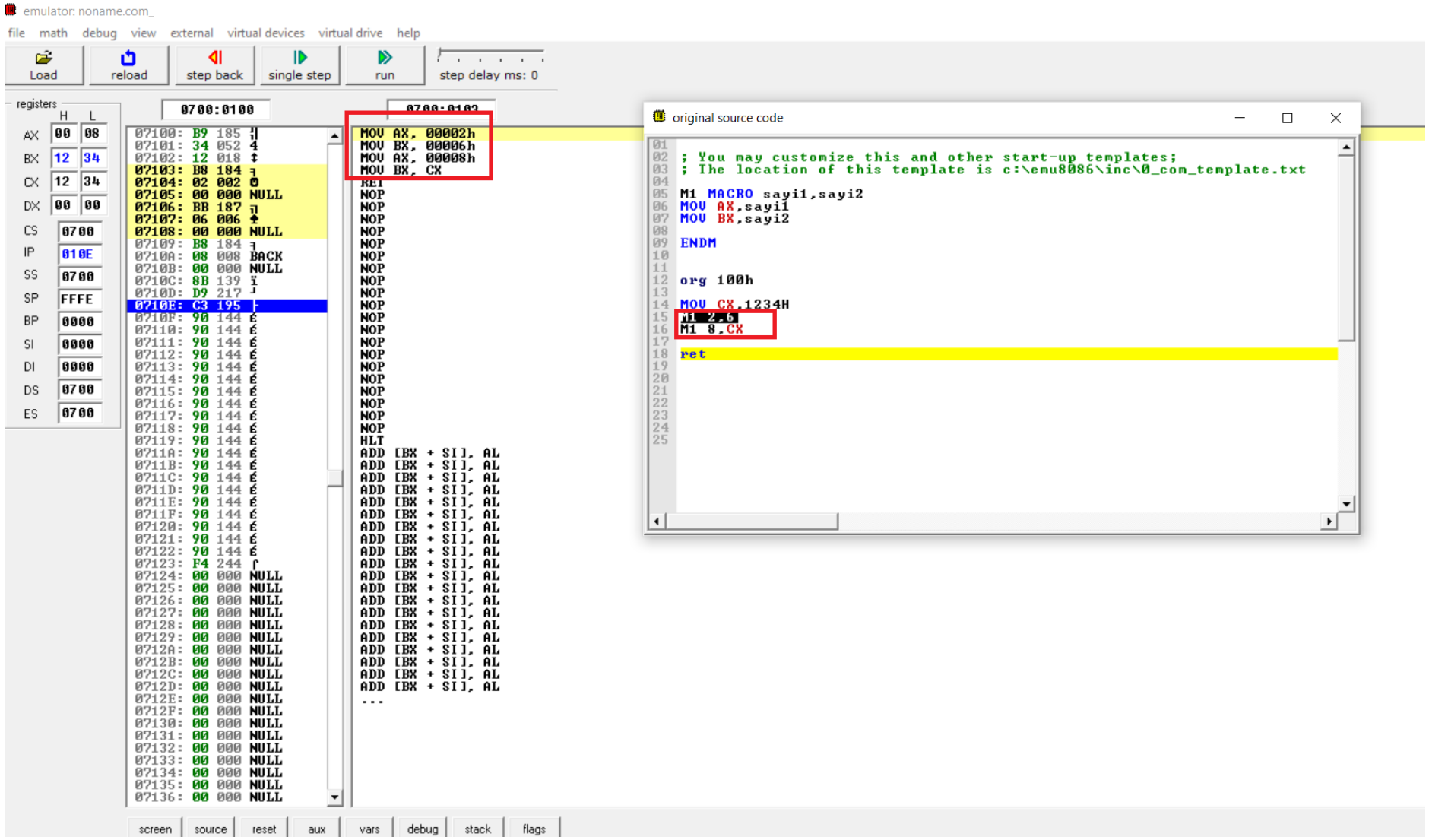
- **MACROLAR**

```
M1 MACRO sayi1,sayi2
      MOV AX,sayi1
      MOV BX,sayi2

   ENDM


org 100h

MOV CX,1234H
M1 2,6
M1 8,CX

ret
```

- **Makrolar eğer program içinde çağrılmazsa program kodlarına dahil edilmezler.**
- **Diğer bir ifadeyle kullanıldığı yerde gerçek kodlara çevrilirler**
- **Prosedürlerde kodlar genişletilmez sadece işlem gerçekleştirilir.**
- **Makrolarda ise makro kaç kez çağrılırsa o kadar genişletilir.**
- **Makrolar çağrılmadan önce tanımlanmalıdır**

8086 16-Bit Mikroişlemci

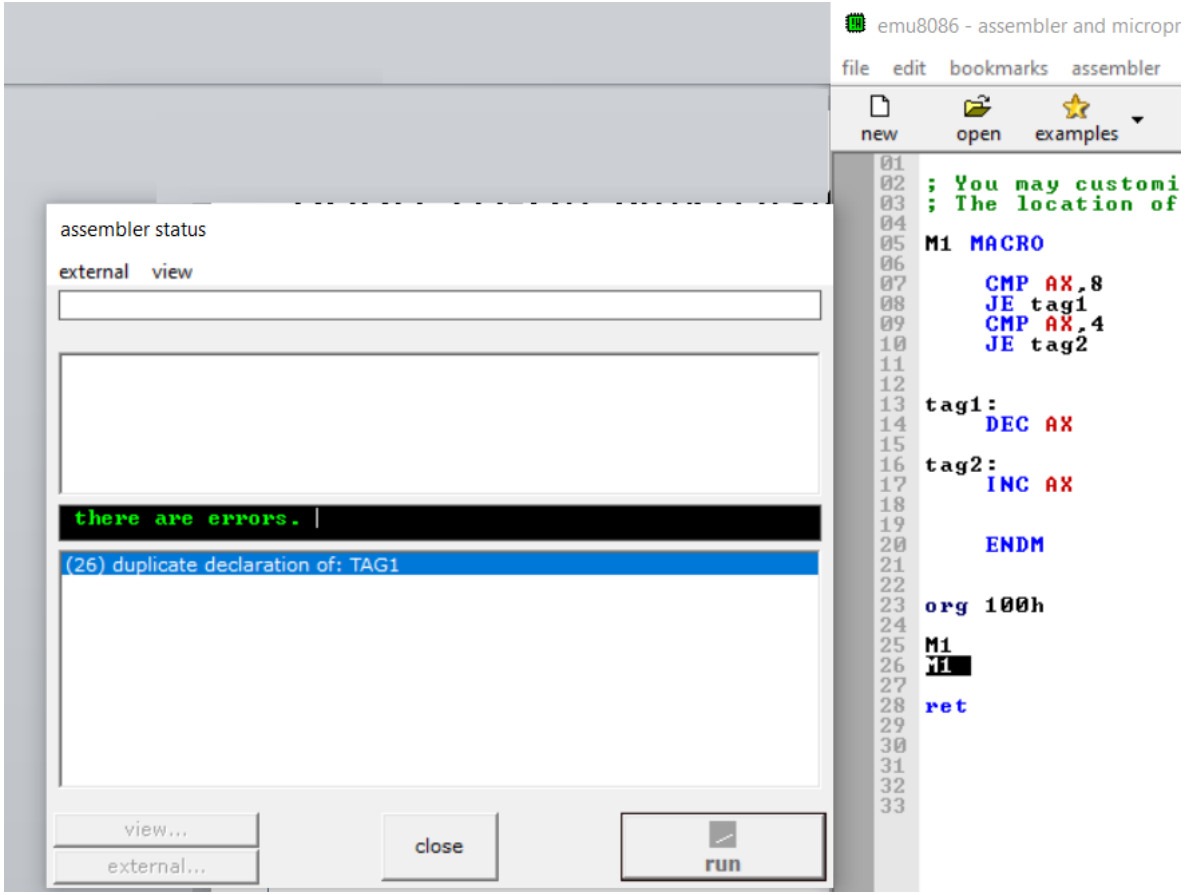
EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
M1 MACRO

LOCAL    tag1,tag2

   CMP AX,8
   JE tag1
   CMP AX,4
   JE tag2


tag1:
   DEC AX

tag2:
   INC AX


   ENDM


org 100h

M1
M1

ret
```



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU 5: kaynak db -2,1,4,-9 dizisini hedef dizisine makro yardımıyla kopyalayan 8086 Assembly kodunu yazınız

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
; multi-segment executable file template.

data segment     ;DS:[SI]
   kaynak db 2,1,4,-9
ends

stack segment
   dw   128  dup(0)
ends

extra segment ;ES:[DI]
   hedef db 4 dup(0)
ends



M1 MACRO
      MOVSB
   ENDM


code segment
start:
; set segment registers:
   mov ax, data
   mov ds, ax
   mov ax, extra
   mov es, ax

   ; add your code here

   lea SI, kaynak
   lea DI, hedef
   MOV CX,4



   dongu:
   M1
   loop dongu




   mov ax, 4c00h ; exit to operating system.
   int 21h
ends

end start ; set entry point and stop the assembler.
```