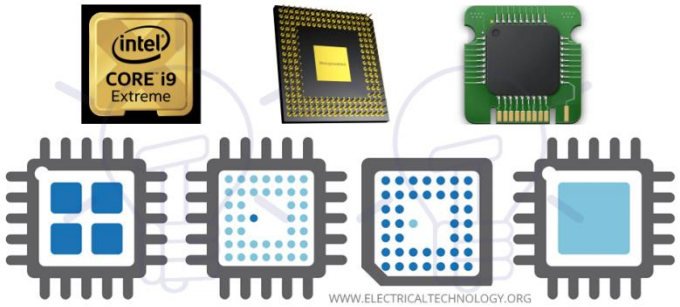


MİKROİŞLEMCİLER

Dr. Meltem KURT PEHLİVANOĞLU

W-11

MİKROİŞLEMCİLER

Digital Logic +

Digital Design +

Computer Architecture +

Microprocessors +

Microcontrollers +

Assembly Language Programming(8086)

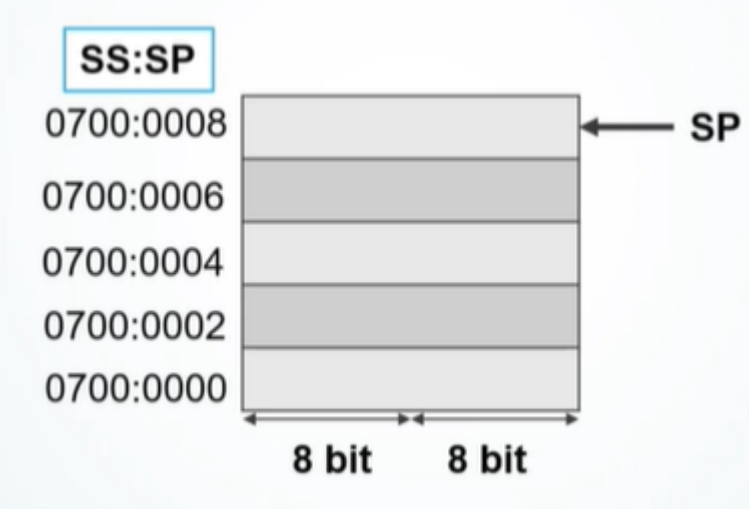
8086 16-Bit Mikroişlemci

- 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

- SP Stack için ayrılan alanın en başını işaret eder.
- Küçük değerlikli 8-bit önce olmak üzere Stack e yerleştirme yapılır
- PUSH Stack e yazma
- POP Stack ten bilgi alma

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **PUSH** operand1
  - SP =SP-2
- **POP** operand1
  - SP=SP+2

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **PUSHA :** operand almaz, tüm genel amaçlı (AX,BX,CX,DX,DI,SI,BP,SP) registerların değerlerini stack üzerine yükler. SP değeri komut kullanılmadan önceki haliyle yüklenir.
- Bu komuttan sonra herhangi bir stack islemi yapılmamalıdır

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **POPA :** operand almaz, genel amaçlı (AX,BX,CX,DX,DI,SI,BP) registerların değerlerini stack den alıp yükler. **SP değeri geri yüklenmez**
- **SP değeri geri yüklenmiyor çünkü eğer geri yüklense SP kaldığı yerden farklı bir konuma geçer. SP değeri geri yüklenmediği için, SP ın kaldığı yer kaybolmamış olur.**
- Bu komuttan sonra herhangi bir stack islemi yapılmamalıdır

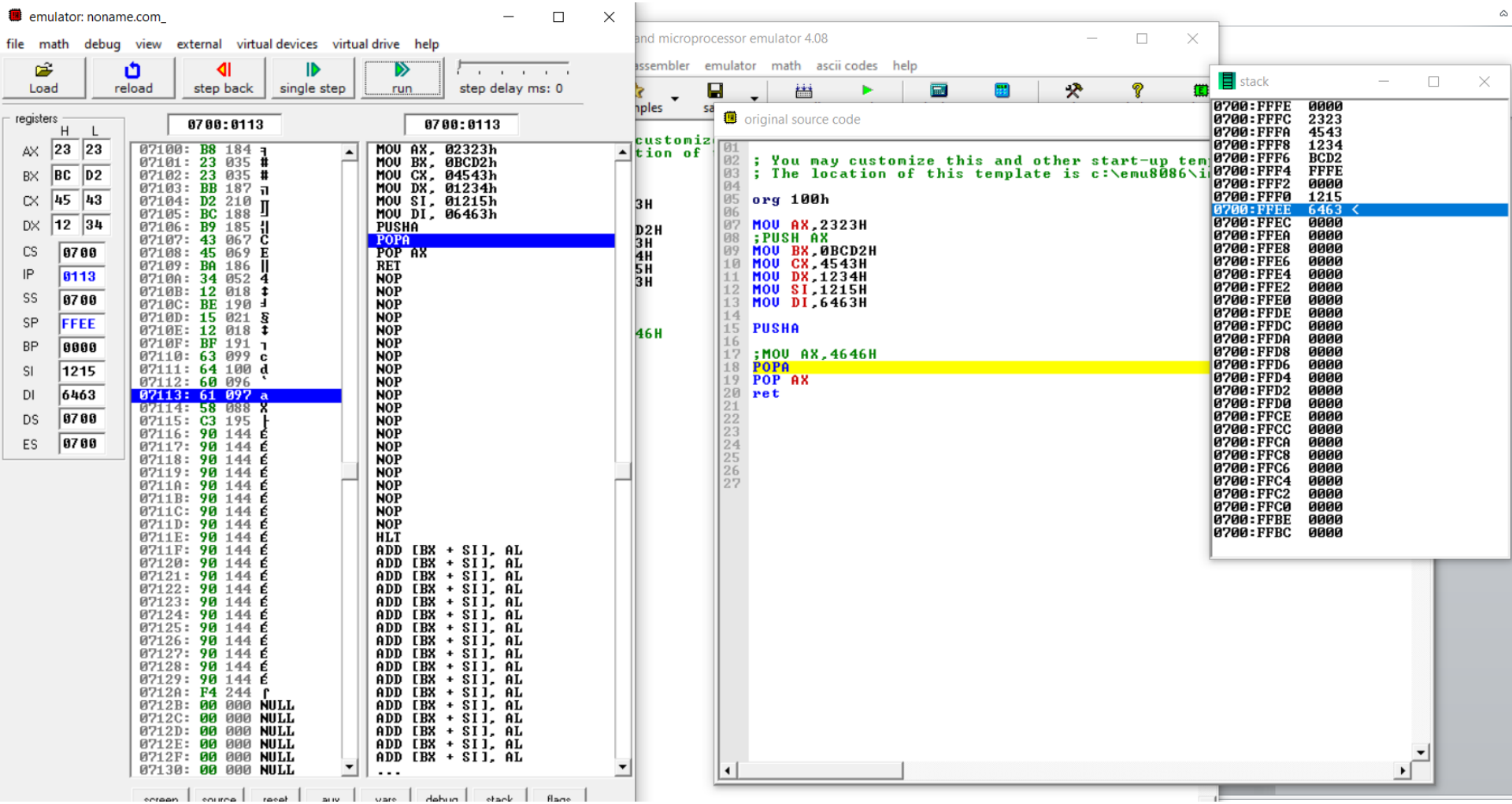
8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h

MOV AX,2323H
;PUSH AX
MOV BX,0BCD2H
MOV CX,4543H
MOV DX,1234H
MOV SI,1215H
MOV DI,6463H

PUSHA

;MOV AX,4646H
POPA
POP AX ; burda Stack islemi yapildigi icin SP konumunun degistigi gozlemlenebilir
ret
```

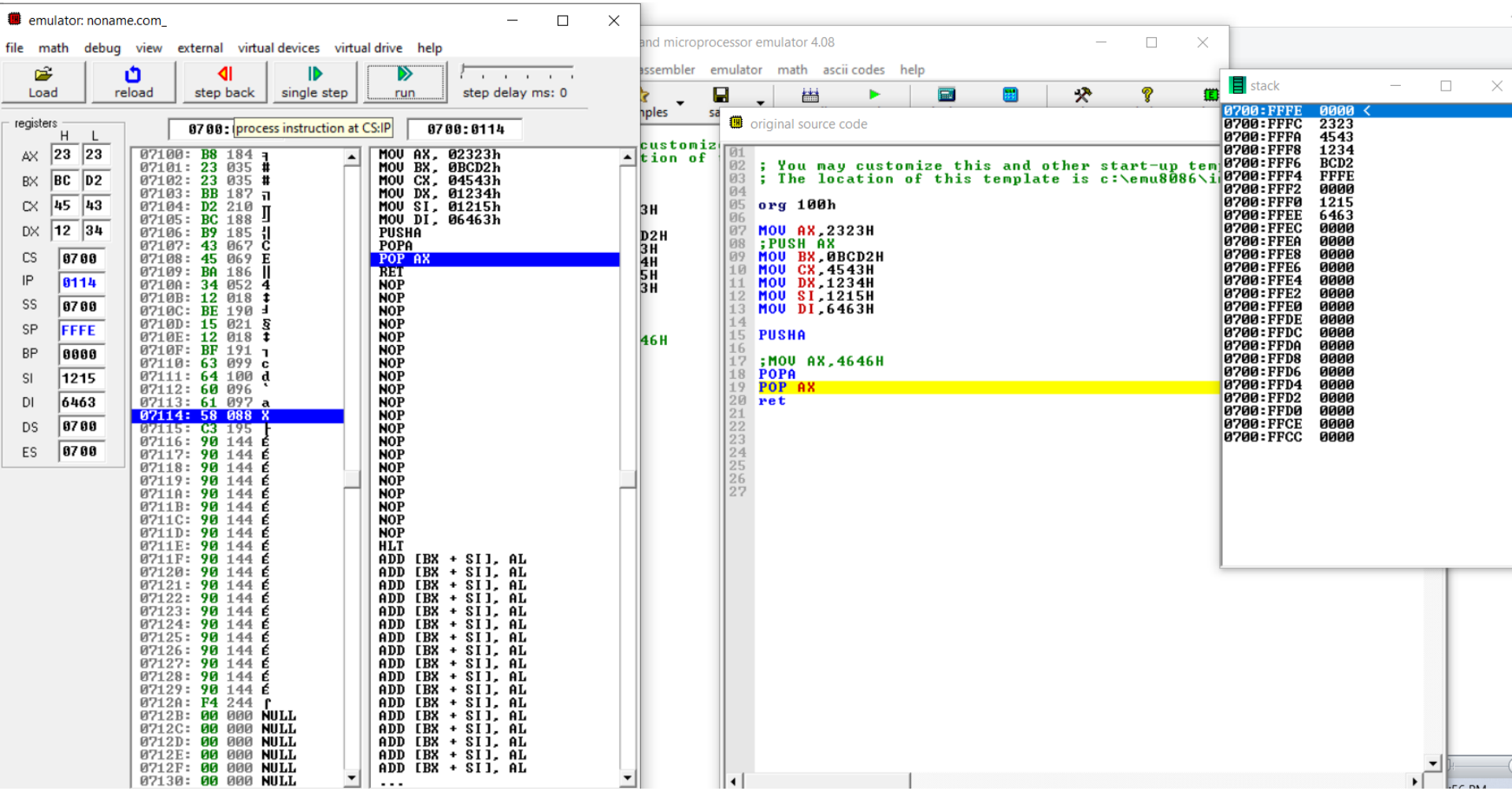
8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR



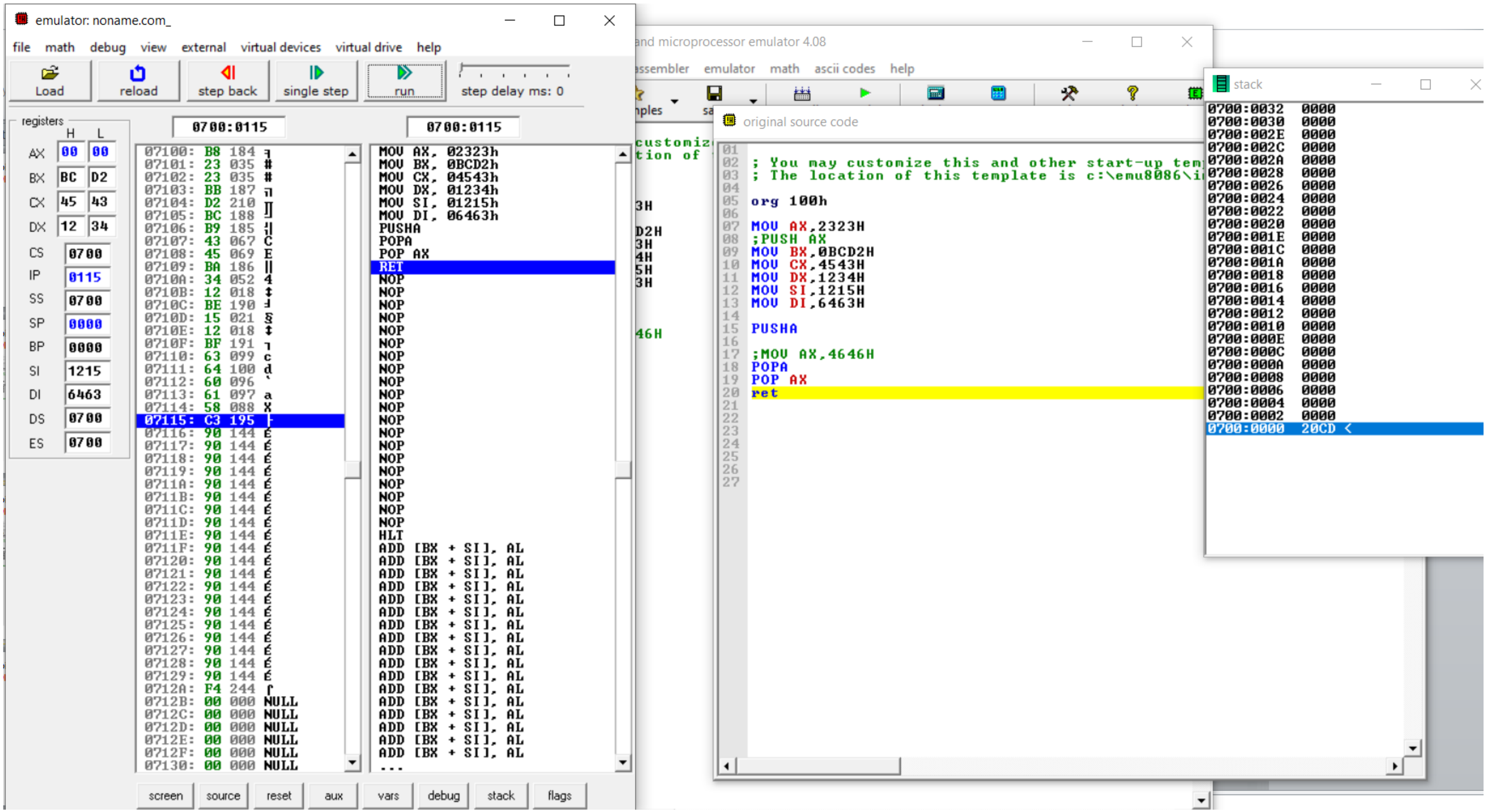
8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

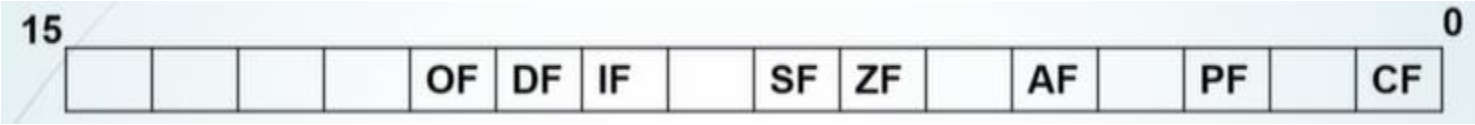
EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

8086 16-bit Flag Register



Overflow Flag **(OF) – Taşma Bayrağı**
Direction Flag **(DF) – Yön Bayrağı**
Interrupt Enable Flag **(IF) – Kesme Aktif Bayrağı**
Sign Flag **(SF) – İşaret Bayrağı**
Zero Flag **(ZF) – Sıfır Bayrağı**
Auxiliary Carry Flag **(AF) – Yardımcı Elde Bayrağı**
Parity Flag **(PF) – Benzerlik(Eşlik) Bayrağı**
Carry Flag **(CF) – Elde Bayrağı**

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

Bayrak Değişim Komutları:

- **STC(Set Carry flag)**: Carry flag aktif duruma=1 getirir.
- **CLC(Clear Carry flag):** Carry flag pasif duruma=0 getirir.
- **CMC(Complement Carry flag):** Carry flag aktifse pasif, pasifse aktif yapar.

- **STD(Set Direction flag):** Direction flag aktif duruma=1 getirir
- **CLD(Clear direction flag):** Direction flag pasif duruma=0 getirir

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **STI (Set Interrupt enable flag):** Interrupt flag aktif duruma=1 getirir (işlemci donanımsal kesmelere izin verilir, varsayılan 1 gelir)
- **CLI (Clear Interrupt enable flag):** Interrupt flag pasif duruma=0 getirir
- **LAHF(Load AH from 8 low bits of Flags register):** Flag registerın düşük değerli 8 bitini AH kaydedicisine aktarır. Bayrak değerleri değişmez. 1,3,5. bitler rezerve edilmiş

AH bit:

SF	ZF	0	AF	0	PF	1	CF
7	6	5	4	3	2	1	0

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **LAHF(Store AH register into low 8 bits of Flags register):** AH içinde bulunan değer flag register içine yüklenir

SF	ZF	0	AF	0	PF	1	CF
7	6	5	4	3	2	1	0

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **PUSHF :** Flag register değerlerini stack üzerine yazar.
- operand almaz,
- SP değeri iki azalır SP=SP-2
- Flag register 16-bittir

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **POPF :** Flag register değerini stack üzerinden okur ve yazar.
- operand almaz,
- SP değeri iki artar SP=SP+2
- Flag register 16-bittir

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

ARİTMETİKSEL KOMUTLAR:

- **ADD** operand1,operand2

operand1=operand1+operand2

- **ADC** operand1,operand2

operand1=operand1+operand2+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 alinacak deger
add al,1    ; IF=1 zaten varsayilan geliyor
        ; CF=1 255+1=256  isaretsiz sayilarda tasma oldu
        ; ZF=1 oldu 256-256=0 islem sonucu 0 oldugundan ZF aktif oldu
        ; PF=1 1 bitlerin sayisi 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 toplamasi



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

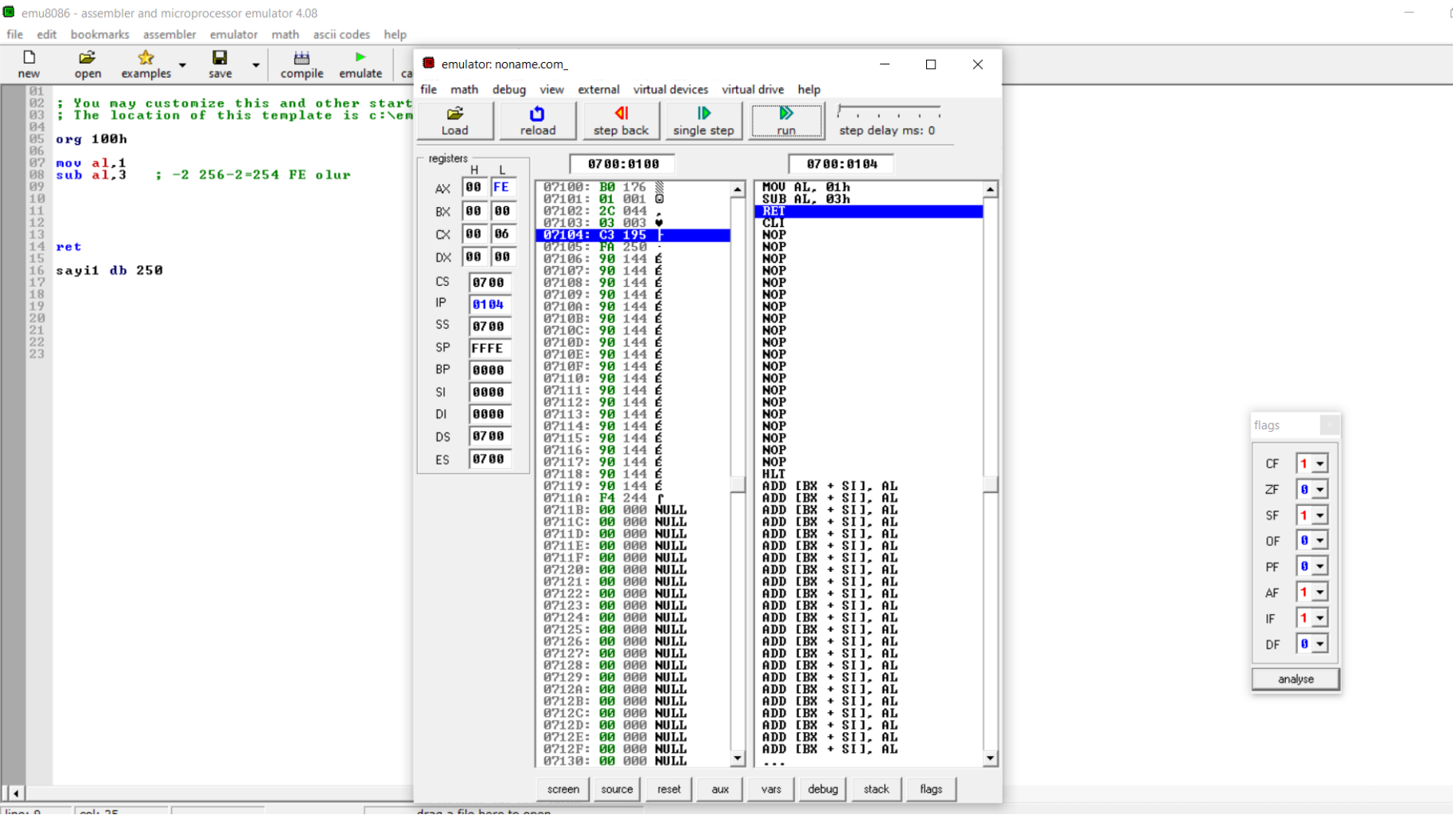
operand1=operand1-operand2-CF

```
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
```

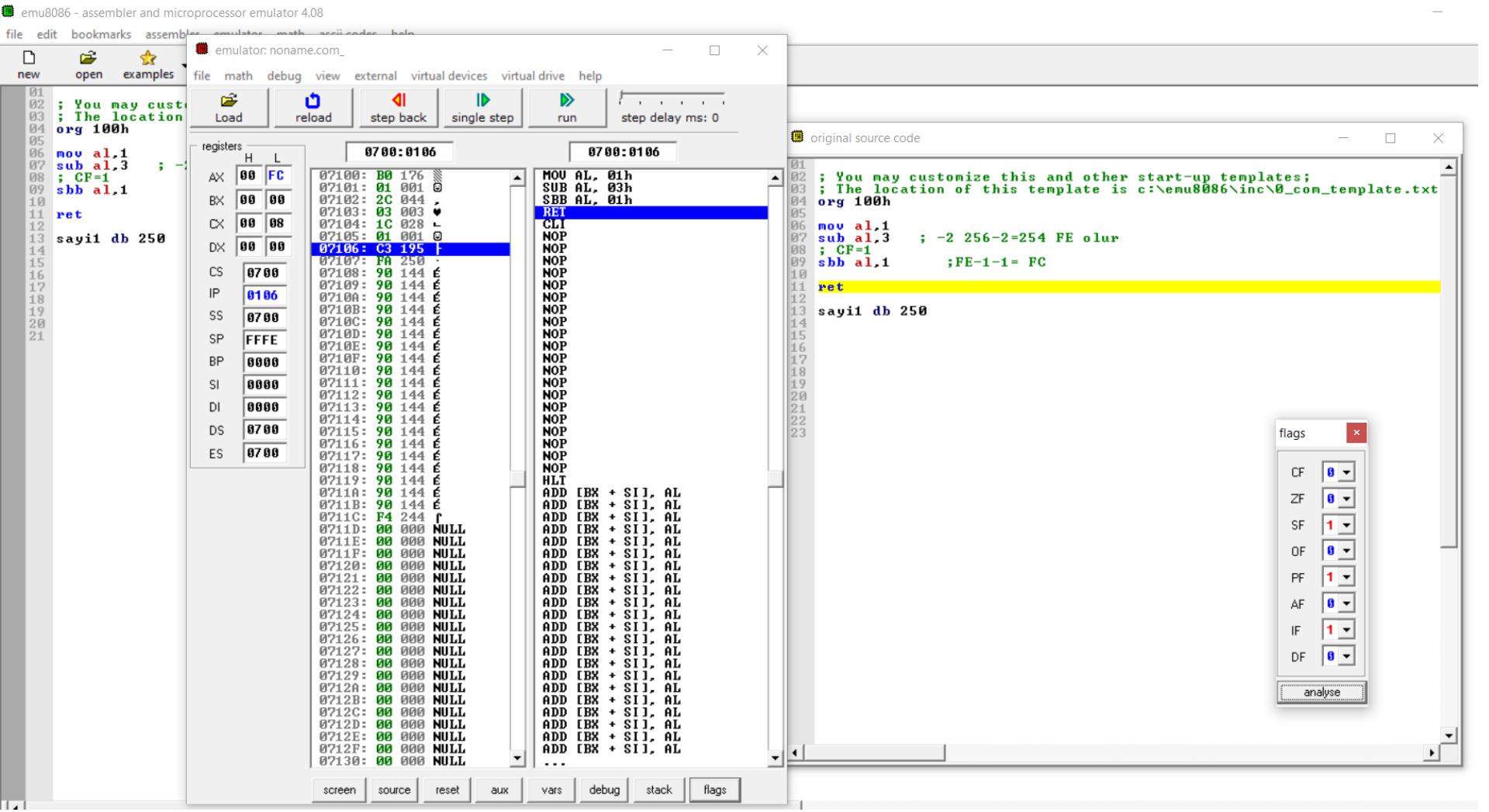
8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **MUL** işaretsiz sayılarda çarpma

AX = AL * operand1 (8-bit iki operandın çarpımı: sonuc AX te tutulur, operand1 register veya ram hücresi olabilir)

(DX AX) = AX * operand1 (16-bit iki operandın çarpımı: DX ve AX te tutulur 16-biti aşma durumuna karşı DX te kullanılır, operand1 register veya ram hücresi olabilir)

- Çarpma sonucunun yüksek değerli (AH=0 (8-bit için), DX=0 (16-bit için)) kısmı 0 olduğu zaman CF=OF=0 olur

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h

mov al,6
mov bl,2   ; ikinci carpilacak degeri baska registera atamalisiniz
mul bl     ; carpim sonucu AX te AX=000C

mov al,255
mov bl,2
mul bl     ; AX=01FE olur unsigned=510

mov ax,2
mul [sayi1] ; sonucu DX AX te gorurum    DX=0000 AX=0200 (unsigned= 512)

mov ax,65535 ; yani max 16-bitle ifade edebilecegim deger FFFF
mul sayi1    ; 65535*256=16776960  = 00FFFF00h    DX=00FF AX=FF00

ret
sayi1 dw 256
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **IMUL** işaretli sayılarda çarpma

AX = AL * operand1 (8-bit iki operandın çarpımı: sonuc AX te tutulur, operand1 register veya ram hücresi olabilir)

(DX AX) = AX * operand1 (16-bit iki operandın çarpımı: DX ve AX te tutulur 16-biti aşma durumuna karşı DX te kullanılır, operand1 register veya ram hücresi olabilir)

- Çarpma sonucu 8-bit (-128,+127) 16-bit (-32768, +32767) olduğu zaman CF=OF=0 olur

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h

mov ax,-2
mov bx,20
imul bx   ; DX=FFFF AX=FFD8 -40



ret
sayi1 dw 256
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **DIV** işaretsiz sayılarda bölme

AL= AX / operand1 (8-bit sonuc: AH=kalan operand1 register veya ram hücresi olabilir)

AX= (DX AX) /operand1 (16-bit sonuc: DX=kalan, operand1 register veya ram hücresi olabilir)

8086 16-Bit Mikroişlemci
EMU 8086-MICROPROCESSOR EMULATOR

- **IDIV** işaretli sayılarda bölme

AL= AX / operand1 (8-bit sonuc: AH=kalan operand1 register veya ram hücresi olabilir)

AX= (DX AX) /operand1 (16-bit sonuc: DX=kalan, operand1 register veya ram hücresi olabilir)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h

mov ax,115
div [sayi1] ; sonuc AL de kalan AH da tutulur   AH=0 ise tam bolunuyor


mov dx,0001h
mov ax,1200h    ;00011200 sayisi  70144 decimal
mov bx,5h    ; 70144/5 = 14028 sayisi HEX olarak 36CC olur (AX=36CC)    Kalan=4
(DX=0004)
div bx

mov ax,-20
mov bl,10
idiv bl

ret
sayi1 db 5
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **CBW:** 8-bitlik değeri 16-bitlik değere genişletir

Operand almaz, yüksek değerli 8. biti AH içine yayar

- **CWD:** 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
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

MANTIKSAL İŞLEMLER

- **AND** operand1, operand2 : mantıksal ve işlemi yapar, sonuc operand1 de tutulur

CF=0, OF=0 olur ZF, SF, PF işlem durumuna göre değişir

```
org 100h

; b karakterini B ye donusturmek  b=98  B=66
; 1. durum: -32 = 11100000 ile toplanabilir   (98-32=66)
; 2. durum: 32 ile maskeleme 00100000 diger bir ifadeyle (11011111) degeriyle and leriz

mov al, ‘b'
mov bl,11100000b
add al,bl ; deger ilk operand olan AL de tutulur

;2. durum
mov cl, ‘b'
and cl,11011111b  ; deger ilk operand olan CL de tutulur

ret
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **OR** operand1, operand2 : mantıksal veya işlemi yapar, sonuc operand1 de tutulur

CF=0, OF=0 olur ZF, SF, PF işlem durumuna göre değişir

```
org 100h

; B karakterini b ye donusturmek    B=66   b=98
; 1. durum: 32 = 00100000 ile toplanabilir   (66+32=98)
; 2. durum: 32   00100000 OR lanir


mov al, 'B'
mov bl,00100000b
add al,bl ; deger ilk operand olan AL de tutulur


;2. durum

mov cl, 'B'
or cl,00100000b   ; deger ilk operand olan CL de tutulur


ret
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **XOR** operand1, operand2 : mantıksal XOR işlemi yapar, sonuc operand1 de tutulur

CF=0, OF=0 olur ZF, SF, PF işlem durumuna göre değişir

```
org 100h

mov bl,00100000b
xor bl,10101001b

ret
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **NOT** operand1: operand1 deki değerin bitlerinin tersini alır

Hiçbir bayrağa etki etmez.

```
org 100h


mov bl,00100000b
not bl


ret
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- **TEST** operand1,operand2: operand1,operand2 mantıksal AND işlemine tabi tutulur, ancak sonuç herhangi bir yerde saklanmaz, sadece bayraklar etkilenir

CF=0, OF=0 olur ZF, SF, PF işlem durumuna göre değişir

```
org 100h


mov al,10000101b
test al,00000001b  ; 10000101 AND 00000001 = 00000001    ZF=0
test al,00000010b  ; 10000101 AND 00000010 = 00000000    AL deki deger 0 oldugu icin ZF=1
                   ; islem sonucunda bulunan 1 bitlerinin sayisi cift oldugundan PF 1 olur

ret
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU: İşaretsiz sayilar= 2,4,6,3 dizisindeki elemanların her birinin küpünü alan ve bu elemanları kup dizisine (word tipinde) yazdıran 8086 Assembly kodunu yazınız

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h

MOV CX,4
MOV SI,0
MOV DI,0


dongu:

MOV AL,0
MOV BL,0
MOV AL, [sayilar+SI]
MOV BL,AL   ; BX= AX
MUL BL  ; AX= AX*AX
MUL BL  ; AX= AX*AX*AX
MOV kup+DI,AX

INC SI
ADD DI,2

LOOP dongu

ret

sayilar db 2,4,6,3
kup dw 4 dup(?)
```

```
org 100h


MOV CX,4
MOV SI,0
MOV DI,0


dongu:

MOV AL,0
MOV BL,0
MOV AL, [sayilar+SI]
MOV BL,AL   ; BX= AX
IMUL BL  ; AX= AX*AX
IMUL BL  ; AX= AX*AX*AX
MOV kup+DI,AX

INC SI
ADD DI,2

LOOP dongu

ret

sayilar db -2,4,6,3   ; isaretli eleman oldugunda imul kullanilir
kup dw 4 dup(?)
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU: sayilar= 2,4,6,-2 dizisindeki elemanların her birinin indis değerlerini elemandan çıkararak sonucu fark dizisine yazan 8086 Assembly kodunu yazınız

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h


MOV CX,4
MOV SI,0
MOV DL,0

dongu:

MOV AL,0
MOV BL,0
MOV AL, [sayilar+SI]
MOV BL,DL
SUB AL,BL  ; sonuc AL de
MOV fark+SI,AL

INC SI
INC DL
LOOP dongu

ret

sayilar db 2,4,6,-2
fark db 4 dup(?)
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU: cumle='buyukharfyazilacak' dizisindeki tüm elemanları büyük harfe çevirip bunu buyukharf dizisine aktaran 8086 Assembly kodunu yazınız

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h

MOV CX,18
MOV SI,0

dongu:

MOV AL, [cumle+SI]
AND AL,11011111b     ;maskeleme islemi
MOV buyukharf+SI,AL

INC SI

LOOP dongu

ret

cumle db 'buyukharfyazilacak'
buyukharf db 18 dup(?)
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU: sayilar= 10,20,30,40 dizisindeki elemanların her birinin sadece son 4 bitindeki değerleri alıp bunları sayilar2 dizisine yazan 8086 Assembly kodunu yazınız

Örn: 20= 0001 0100→ 0100 (4) olacak

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h

MOV CX,4
MOV SI,0

dongu:
MOV AL, [sayilar+SI]
AND AL,00001111b
MOV sayilar2+SI,AL
INC SI
LOOP dongu

ret

sayilar db 10,20,30,40
sayilar2 db 4 dup(?)
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- SORU: sayilar= 10,20,30,40 dizisindeki elemanları sayilar2= 3,4,7,6 dizisindeki sayılara bölen ve sadece kalanları kalan dizisine yazan 8086 Assembly kodunu yazınız

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h
MOV CX,4
MOV SI,0

dongu:
MOV AX,0
MOV BX,0
MOV AL, [sayilar+SI]
MOV BL, [sayilar2+SI]
DIV BL ; kalan AH icinde
MOV [kalan+SI],AH
INC SI
LOOP dongu

ret

sayilar db 10,20,30,40
sayilar2 db 3,4,7,6
kalan db 4 dup(?)
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

SORU: Stack üzerine **veri1=BCDEh**, **veri2=AB03h** değerlerini yazın. Daha sonra bu verileri **AX** registerına çekip veri1 ve veri2 nin **düşük değerlikli** 8-bitini **dusuk** dizisine, **yüksek değerlikli** 8-bitini **yuksek** dizisine yazınız.

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

```
org 100h

PUSH 0BCDEh
PUSH 0AB03h


MOV CX,2
MOV SI,0

dongu:
POP AX
MOV buyuk+SI,AH
MOV kucuk+SI,AL
INC SI
LOOP dongu



ret
buyuk db 2 dup(?)
kucuk db 2 dup(?)
```

```
org 100h

MOV AX,veri1
PUSH AX
MOV AX,veri2
PUSH AX


MOV CX,2
MOV SI,0
; dongulu kullanim
dongu:
POP AX
MOV buyuk+SI,AH
MOV kucuk+SI,AL
INC SI
LOOP dongu



ret

veri1 dw 0BCDEh
veri2 dw 0AB03h
buyuk db 2 dup(?)
kucuk db 2 dup(?)
```

```
org 100h

MOV AX,veri1
PUSH AX
MOV AX,veri2
PUSH AX

; dongusuz yapmak istersem


POP AX
MOV buyuk[0],AH
MOV kucuk[0],AL



POP AX
MOV buyuk[1],AH
MOV kucuk[1],AL



ret

veri1 dw 0BCDEh
veri2 dw 0AB03h
buyuk db 2 dup(?)
kucuk db 2 dup(?)
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR