

# MİKROİŞLEMCİLER (BLM202)

## HAFTA - 7

Dr. Bilgin YAZLIK, RTTP, PMP



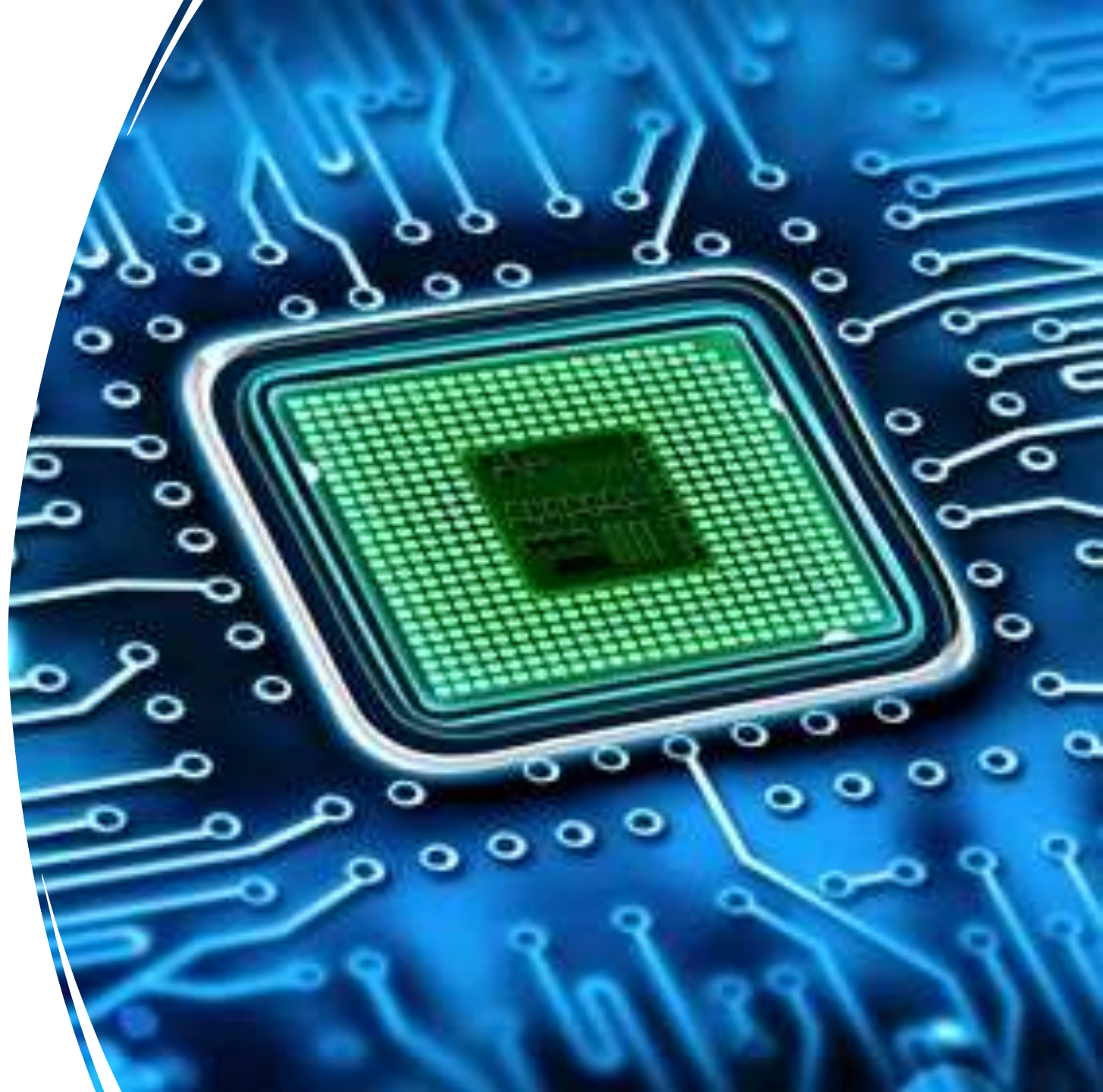
**BİLGİSAYAR MÜHENDİSLİĞİ**



# 7. HAFTA

---

- Veri Aktarım Komutları
- Adresleme Modları



# LEA ÖRNEK

---

- VAR1 DB 22H
- LEA BX, VAR1
- MOV BYTE PTR[BX], 44H
- MOV AL, VAR1
- END





# XCHG(Exchange)

---

- XCHG reg,reg
- XCHG reg,mem
- XCHG mem,reg
- XCHG işlenenlerin değerlerinin değiş tokuş edilmesini sağlayan komuttur.
- Aynı sonucu, biraz daha uzun sürede ek bir yazmaç veya bellek alanı kullanarak daha fazla kod üreterek MOV komutları ile elde etmek de mümkündür.
- Ancak assembly dili mevcut kaynakların sınırlı olduğu durumlarda çözümler üretebilen bir dil olup bu gibi durumlarda XCHG komutu tercih edilmelidir.





# ÖRNEK

---

- **XCHG AX,BX;** AX yazmacının sahip olduğu değer ile BX yazmacının sahip olduğu değer değiştirilir.
- **XCHG AX,Mydata[SI];** Burada ise AX yazmacının değeri Mydata isimli bellek adresinin SI'ncı adresinden başlayan word yer değiştirmektedir. Mydata word olarak tanımlanmalıdır.


# ÖRNEK

- `#make_COM#`
- `;` COM file is loaded at CS:0100h
- `ORG 100h`
- `X DW 35`
- `mov ax,10`
- `xchg ax,x ; x, ax olur, ax de x olur`
- `hlt`

## the output file type directives:

`#make_com# #make_bin# #make_boot# #make_exe#` You can insert these directives in the source code to specify the required output type for the file. Only if compiler cannot determine the output type automatically and it when it cannot find any of these directives it may ask you for *output type* before creating the file.

There is virtually no difference between how `.com` and `.bin` are assembled because these files are raw binary files, but `.exe` file has a special header in the beginning of the file that is used by the operating system to determine some properties of the executable file.



# XLAT (Translate byte)

- XLAT
- Doğrudan işleneni olmayan bir komuttur. AL yazmacını, başlangıcı DS:BX yazmaçları ile belirlenen adresteki bir tablo içerisinde indis olarak kullanmayı sağlar.

# ÖRNEK

---

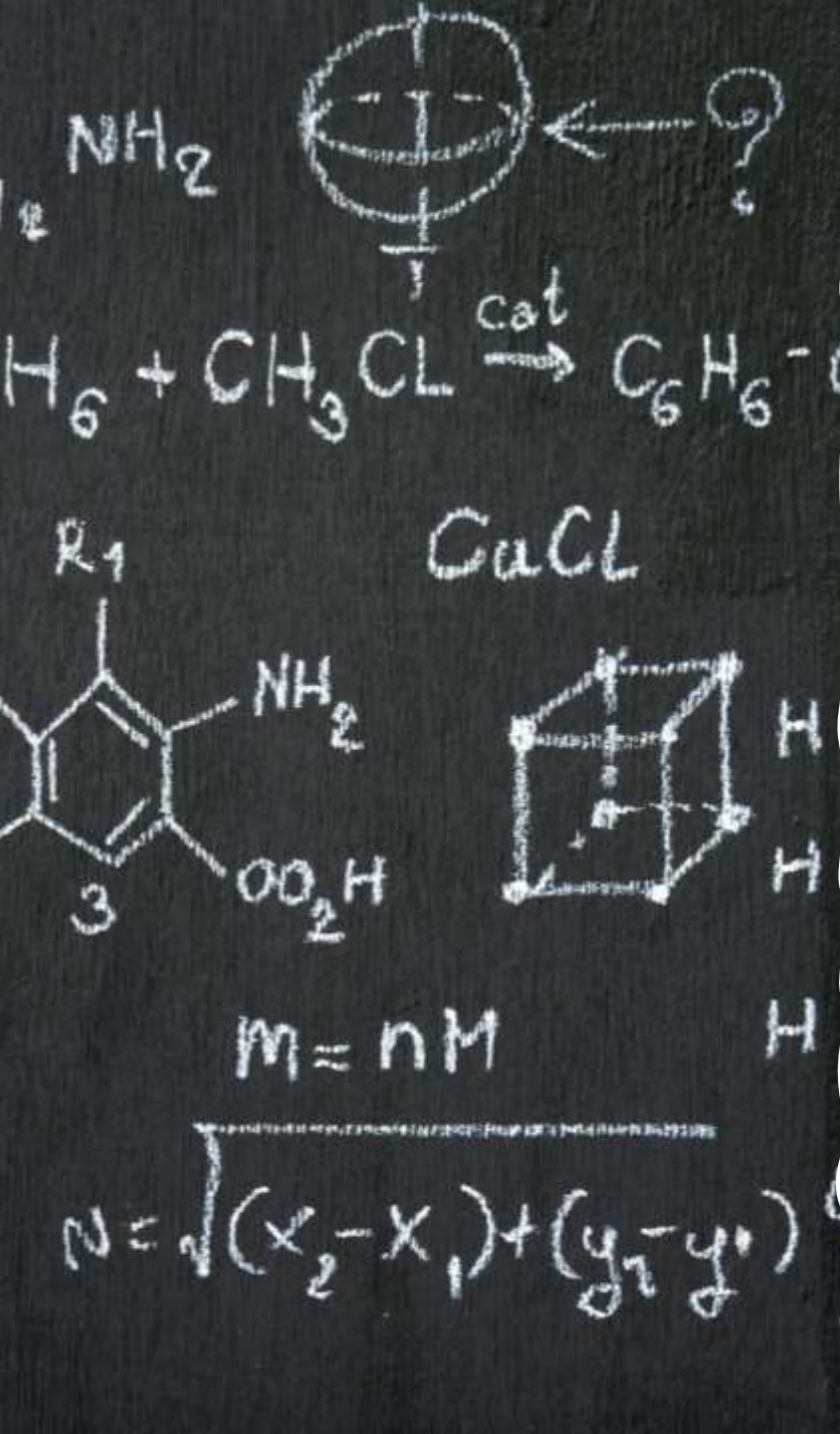
- **LEA BX,Ascii2ebcdic**
- **MOV AL,'A'**
- **XLAT**
- Ascii2ebcdic isimli bellek alanında ASCII ve EBCDIC kodlama sistemi arasındaki dönüşüm yapmayı sağlayan 256 byte uzunluğunda bir dizi olduğunu, dizinin indislerinin ASCII kodlar, içeriğinin ise ASCII kodlara karşılık gelen EBCDIC değer olduğunu düşünelim.
- AL yazmacına 'A' yerleştirilip, XLAT komutu işlendiğinde
- Diziden 'A' harfinin EBCDIC karşılığı alınarak AL yazmacına konacaktır.



# ÖRNEK

---

- #make\_COM#
- ; COM file is loaded at CS:0100h
- ORG 100h
- dizi db 0,8,2,3,1,9,12,7,8,0,10,11,12,13,14
- lea bx,dizi
- mov al,5
- xlat
- hlt
- Not: İlk öge 0 indisten başlar.
- 5. elemanın değeri olan 9 sayısını AL'ye aktarır.



# ADRESLEME MODLARI

- Adresleme modları belleğin nasıl kullanıldığını, belleğe nasıl erişileceğini ve verilerin belleğe nasıl yerleştirileceğini belirler. Sunumda verilen anlatımda kullanılan kelimeler ve anlamları şu şekildedir.
- Register: Kaydedici (Yazmaç)
- Memory: Bellek
- Immediate : Acil veri, doğrudan veri

# Acil Adresleme (Immediate Addressing) (Anlık Adresleme)

- Doğrudan sabit bir değer bir kaydediciye aktarılır. Sabit değerın büyüklüğü ile register uyumlu olmalıdır.

- **Örneğin** 8 bitlik bir kaydediciye 16 bitlik bir değer yüklenemez.

- Genel Kullanımı:

- **KOMUT register, immediate**

- Örnek:

- MOV CL, 16D

- MOV DI, 2ABFH

- MOV AL, 4567H ; Yanlış

## Kaydedici Adresleme (Register Addressing) (Yazmaç Adresleme)

---

- Bu adresleme modunda her iki operand (işlenen) da kaydedicidir.
- Genel kullanımı:
- **KOMUT register, register**
- Örnek:
- MOV AL, BL



# Doğrudan adresleme (Direct addressing)

---

- Doğrudan bir adres değeri kullanılır. Bir adresten bir kaydediciye veri aktarımı gerçekleştirilir. Bir başka ifade ile operandlardan birisi adres belirtir.

- Genel kullanımı:

- KOMUT register, memory veya

- KOMUT memory, register

- Örnek:

- MOV AX, [1000]

- TOPLAM DW 20 ; Burada TOPLAM bir adrestir. 20 sayısı bu adresin içindeki değerdir.

- MOV AX, TOPLAM ; TOPLAM adresindeki değeri AX e at.

- MOV TOPLAM, AX

# Dolaylı adresleme (Indirect addressing) (Kaydediciye dayalı dolaylı adresleme)

- Etkin adres değeri (offset) BX, BP, SI, DI kaydedicilerinden birinde bulunur.
- Genel kullanımı:
  - KOMUT register, [BX/BP/SI/DI] veya
  - KOMUT [BX/BP/SI/DI], register
- Örnek:
  - MOV AX,[SI]
  - MOV BX,1000;
  - MOV [SI], AL
  - TABLO DB 5,9,0,3,-7
  - LEA BX,TABLO
  - MOV AX, [BX] ; Kaydedici dolaylı adresleme var

# İndisli adresleme (Indexed addressing)

---

- Dolaylı adreslemede köşeli parantez içinde kalan BX/BP/SI/DI kaydedicilerine bir indis değeri eklenerek kullanılır.

- Genel kullanımı:

- KOMUT register, [BX/BP/SI/DI+indis] veya

- KOMUT [BX/BP/SI/DI+indis], register

- **Örnek:**

- MOV AX, [SI+4]

- MOV CX, [DI+7] ;

- MOV AX, [BX+DI]

- MOV AX, [SI-7] ; bu komut yerine MOV AX, [SI]-7 de kullanılabilir.

# DİKKAT

- 1 -)KOMUT memory, memory

Şeklinde bir kullanım geçerli değildir. Bir bellek bölgesinde başka bir bellek bölgesine veri aktarımı yoktur.

- 2-) Sabit bir değer doğrudan Segment Registerine atanamaz.

•MOV DS,1234 ; YANLIŞ

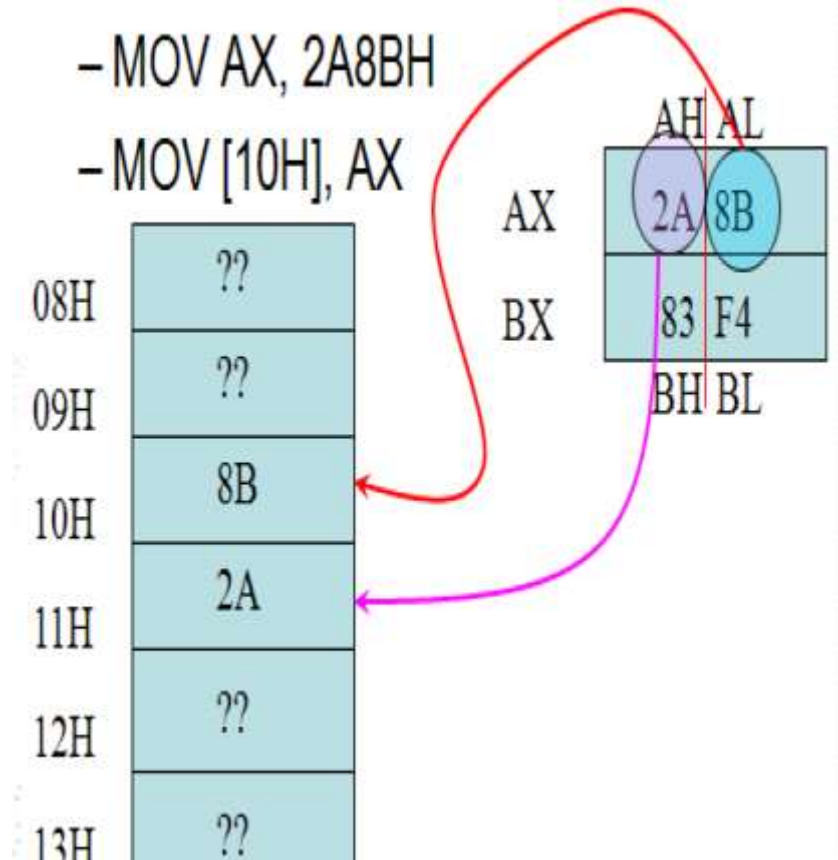
- MOV AX, 1234;DOĞRU
- MOV DS, AX;





# 16 Bitlik bir verinin belleğe yerleşmesi

- Belleğin her bir hücresi 8 bit olduğundan 16 bitlik verinin düşük değerlikli 8 bitlik kısmı adresin düşük değerlikli kısmına, yüksek değerlikli 8 bitlik kısmı adresin yüksek değerlikli kısmına yerleşir. Aşağıda verilen örnekte AX kaydedicisinde 2A8BH değeri yer almaktadır. Düşük değerlikli değer AL de yer almakta ve 10H adresine yerleşmektedir. Yüksek değerlikli kısımda yer alan AH daki değer 2AH değeri ise 11H adresine yerleşmektedir.



# BELLEKTE VERİLERİN SAKLANMASI

---

- Benzer olarak 32bitlik bir veriyi(doubleword) bu şekilde düşünerek yerleştirebilirsiniz.
- Örneğin:
- SAYI DB 1A2F56FFH verisini 100H adresinden itibaren yerleştirecek olursa şu şekilde olur.

Adres	veri
100	FF
101	56
102	2F
103	1A
104	
105	
106	

# Dizi Tanımlaması ve Elemanlarına Erişimi

- DİZİ DB 5, 6, 7, 8, 9, 0, -6, -9, 3 şeklinde tanımlaması yapılır.
- LEA SI, DİZİ ; şeklinde dizinin başlangıç adresini SI kaydedicisine alınır.
- **Örnek:** Bir dizide bulunan elemanların toplamını bulan program kodunu yazınız.

- .MODEL SMALL
- .STACK 64
- .DATA
- DİZİ DB 5, 6, 7, 8, 9, 0, -6, -9, 3, 8
- SONUC DB ?
- .CODE
- ANA PROC FAR
- MOV AX, @DATA; **DS SAYACINI SIFIRLAR**
- MOV DS, AX; **DS SAYACINI SIFIRLAR**
- MOV AL, 0
- MOV CX, 10; **KAÇ DEFA DÖNGÜ OLACAK**
- LEA SI, DİZİ ; DİZİ nin başlangıç ofset adresi SI ya yüklenir
- BAS:
- MOV BL, [SI]
- ADC AL, BL
- INC SI
- LOOP BAS
- MOV SONUC, AL
- MOV AH, 4CH; **PROGRAM SONLANDIRMA**
- INT 21H; **PROGRAM SONLANDIRMA**
- ANA ENDP
- ENDANA

# Kaynaklar

---

- Feza Buzluca, İTÜ Ders Notları, Bilgisayar Mimarisi
- Wikipedia
- Emel Soylu, Kadriye Öz, Karabük Üniversitesi, Mikroişlemciler Ders Notları
- 1) [Bilgisayar Mimarisi – Doç. Dr. Şirzat KAHRAMANLI](#)
- 2) [Ders Notları – Yrd. Doç. Dr. Rifat KURBAN](#)
- Wikipedia
- <https://edukedar.com/difference-between-cisc-and-risc/>
- Dr. B. B. Hegde First Grade College, Kundapura

