ALT SEVİYE PROGRAMLAMA

Hafta 5

Dr. Öğr. Üyesi Erkan USLU

ADRESLEME MODLARI

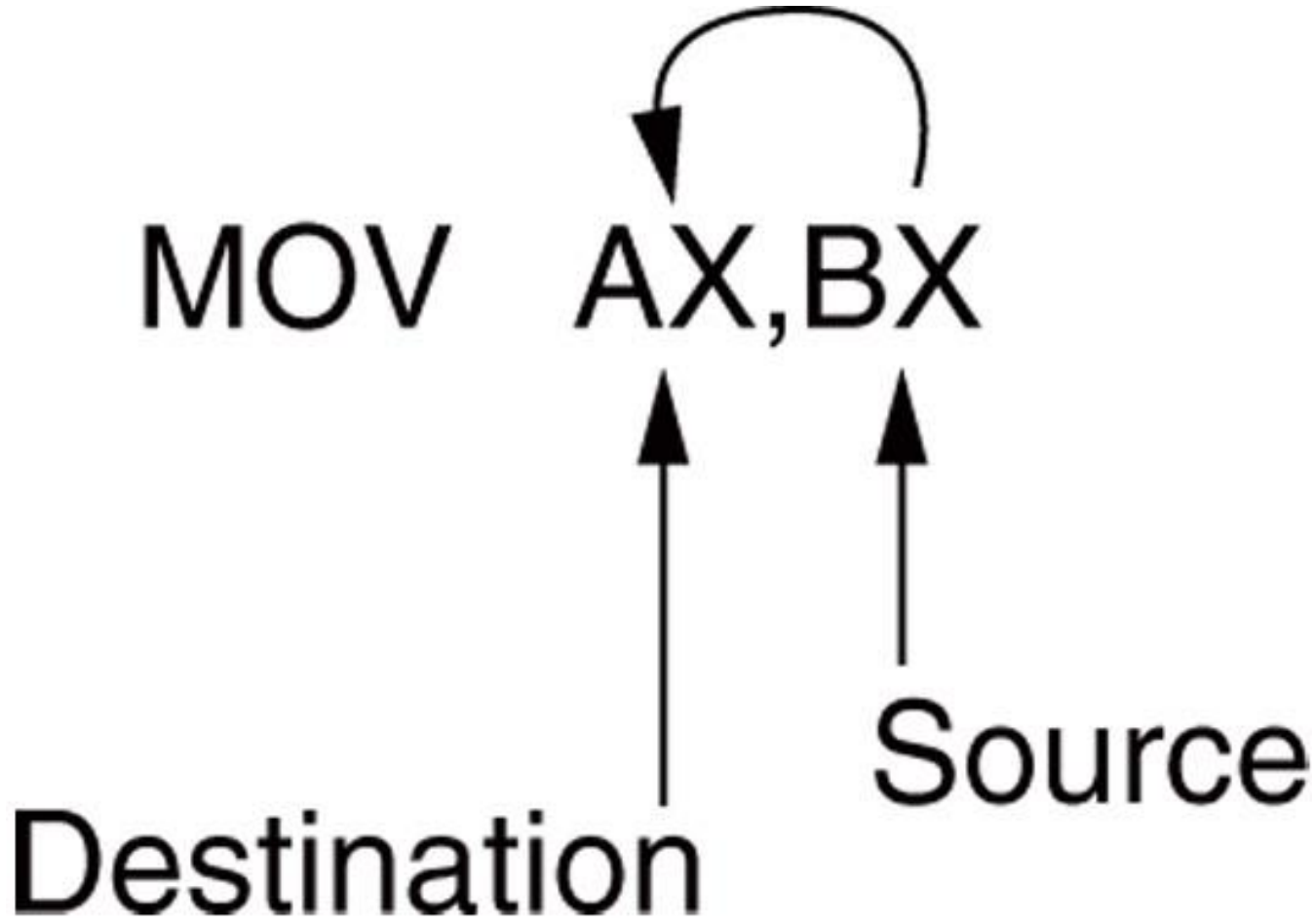
Adresleme Modları

- Verimli assembly programları geliştirebilmek için komutlar ile birlikte kullanılan adresleme modlarının bilinmesi gerekmektedir.

- Üç tip erişim için adresleme modu mevcuttur:
  - Veri adresleme modları
  - Program hafızası adresleme modları
  - Yığın adresleme modları

Veri Adresleme Modları

- 8086 assemly genel yapısı



Veri Adresleme Modları

- 8086 assemly genel yapısı
  - AX ← 1234H
  - BX ← ABCDH
  - MOV AX, BX
  - AX ← ABCDH
  - BX ← ABCDH
- MOV DST, SRC
  - DST ← SRC

Yazmaç Adresleme (Register Addressing)

- Yazmaç Adresleme (Register Addressing)
- 8 bitlik AL, AH, BL, BH, CL, CH, DL ,DH yazmaçları kullanılabilir
- 16 bitlik AX, BX, CX, DX, SP, BP, SI, DI yazmaçları kullanılabilir
- Yazmaç adreslemede kullanılan yazmaç genişlikleri uyumlu olmalıdır

MOV BX, CX

Hemen Adresleme (Immediate Addressing)

- Sabit değer atamayı ifade eder
- 16 bit veya 8 bit sabit değer atama söz konusu olabilir

MOV AL, 0F2H

MOV CX, 100

MOV BL, 01010101B

MOV AH, ‘A’ ;ASCII A karakteri AH yazmacına atanır

Doğrudan Adresleme (Direct Addressing)

- Erişilecek hafıza gözünün doğrudan gösterildiği durumdur

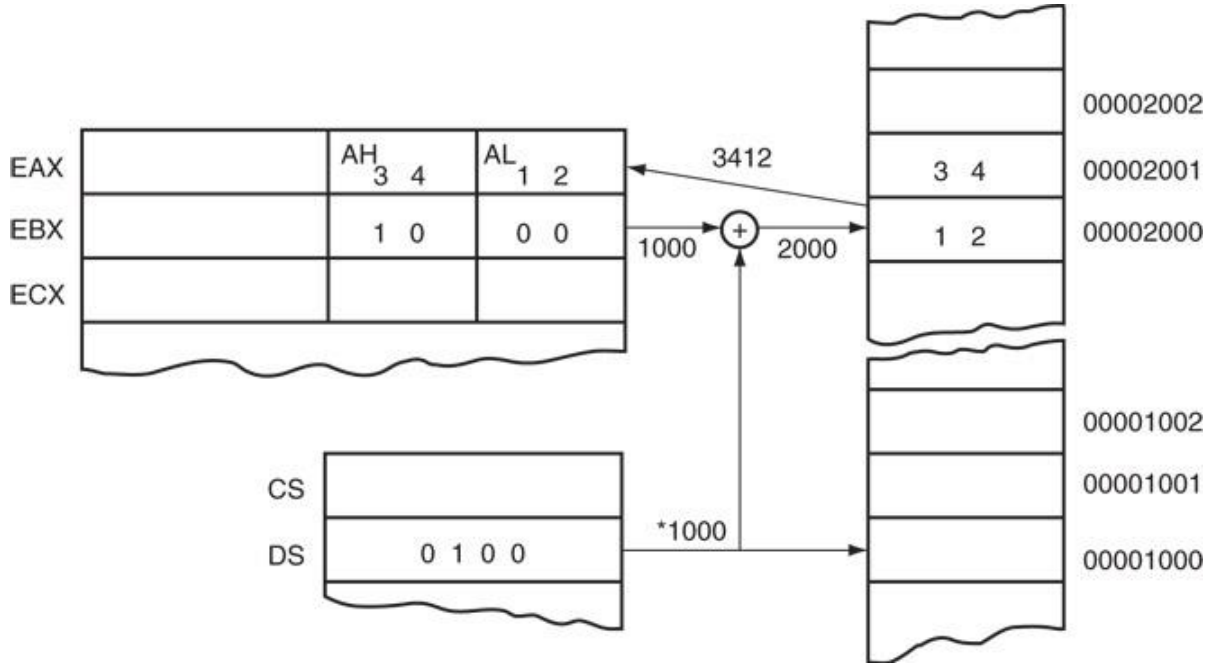
MOV AL, DATA ; DATA bir etiket olup assemler bunu karşılık gelen adres değeri ile değiştirir

MOV BX, [1234H] ; BX ← DS:1234H

Yazmaç Dolaylı Adresleme (Register Indirect Addressing)

- BP (SS ile), BX, DI ve SI (DS ile) yazmaçları ile kullanılabilir
- Hafıza ofset değeri bir yazmaçta saklanır

MOV AX, [BX] ; AX ← DS:BX



Dolaylı Adresleme (Indirect Addressing)

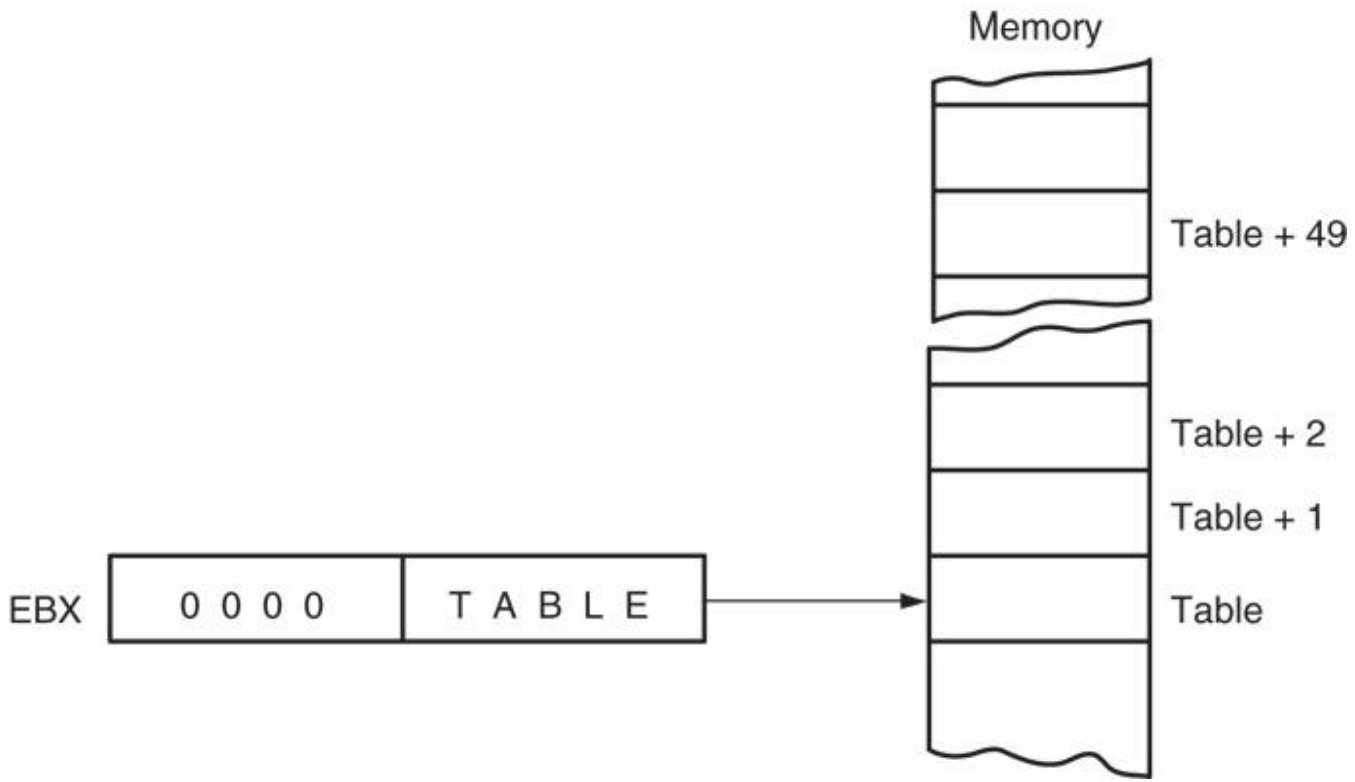
- Dolaylı adreslemelerin bir kısmında BYTE PTR, WORD PTR gibi özel tanımlayıcılar vermek gerekir
- Arttırma komutu olan INC, dolaylı adresleme ile hafızada bir Word mü yoksa Byte değeri mi arttıracağını bilemez

INC WORD PTR [BX]

- Dolaylı adreslemede operandlardan birisi yazmaç ise BYTE PTR, WORD PTR'ye gerek yoktur (NEDEN?)

Yazmaç Dolaylı Adresleme (Register Indirect Addressing)

- Dizi olarak tutulan veriye sıralı erişimde yazmaç dolaylı adresleme kullanımı uygundur



Base+Index Addressing

- Temelde bir dolaylı adresleme modudur
- Base yazmacı (BX veya BP) işlem yapılacak hafıza konumunun başlangıcını göstermek için kullanılır
- Index yazmaçları (DI veya SI) verinin bu başlangıç adresine görece yerini tutmak için kullanılır

MOV DX, [BX+DI]

Yazmaç Göreli Adresleme (Register Relative Addressing)

- Base (BP veya BX) veya Index (DI, SI) yazmaçlarının bir sabit ofset değeri ile kullanılmasını ifade eder

MOV AX, [BX+100H]

Base Relative + Index Addressing

- İki boyutlu veri adresleme için uygudur

MOV AX,[BX + SI + 100H]

Program Hafızası Adresleme Modları

- Program akışı sırasında fonksiyon çağırma, koşullu ve koşulsuz dallanma komutları ile farklı program hafızası adresleme modları kullanılır
- Doğrudan (direct)
- Göreli (relative)
- Dolaylı (indirect)

Doğrudan Program Hafızası Adresleme

- Doğrudan bir program adresine ulaşmak için kullanılır
- Mevcut kod segmentinden farklı bir kod segmentine geçiş sağlayacağı için segmentler-arası bir işlemdir
- Hem CS hem de IP değeri uygun şekilde değiştirilir

JMP 200H:300H ; CS ← 200H, IP ← 300H

CALL 200H:300H

Göreli Program Hafızası Adresleme

- Mevcut IP yazmacı değerine göre hangi program hafızasının adresleneceğini ifade eder
- JMP komutu 1 byte veya 2 byte işaretli sabit değerli operand kabul eder

JMP 100

JMP 0FFH ; IP değeri 1 azalır (NEDEN?)

JMP 1000H

Dolaylı Program Hafızası Adresleme

- CALL ve JMP komutları ile kullanılır

JMP BX

CALL [BX]

Yığın Adresleme Modları

- Tüm yazmaçlardan yığına veri basılabilir
- CS hariç tüm yazmaçlara yığından veri çekilebilir

PUSH CS ; çalışır

POP CS ; assembler hatası verir

Yığın Adresleme Modları

- 8086’da yığın geçici veri saklamak için ve fonksiyonlardan dönüşlerde dönüş adreslerini saklamak için kullanılır
- Yığın LIFO mantığında çalışır (Last In First Out)
- Yığın ile ilgili PUSH ve POP komutları kullanılır
- PUSH yığına WORD basar, POP yığından WORD çeker

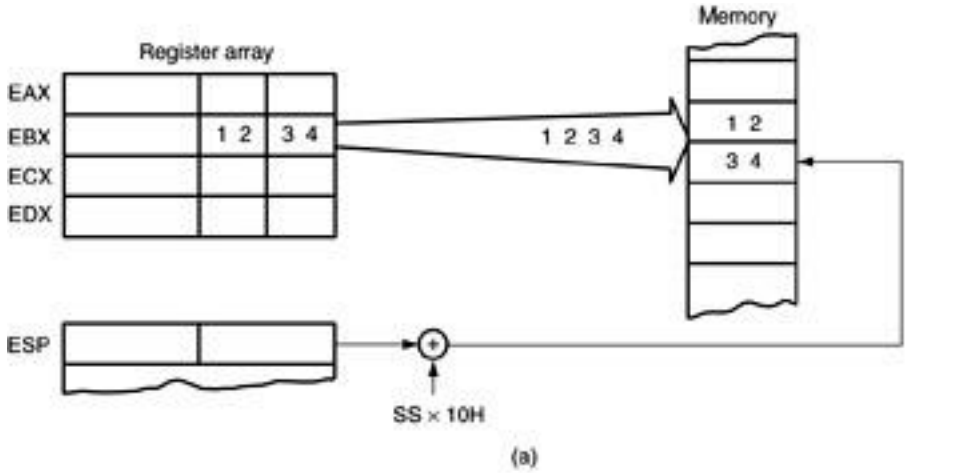
PUSH BL ; (DOĞRU MU?)

- Yığın adresleme için SS:SP ikilisi kullanılır

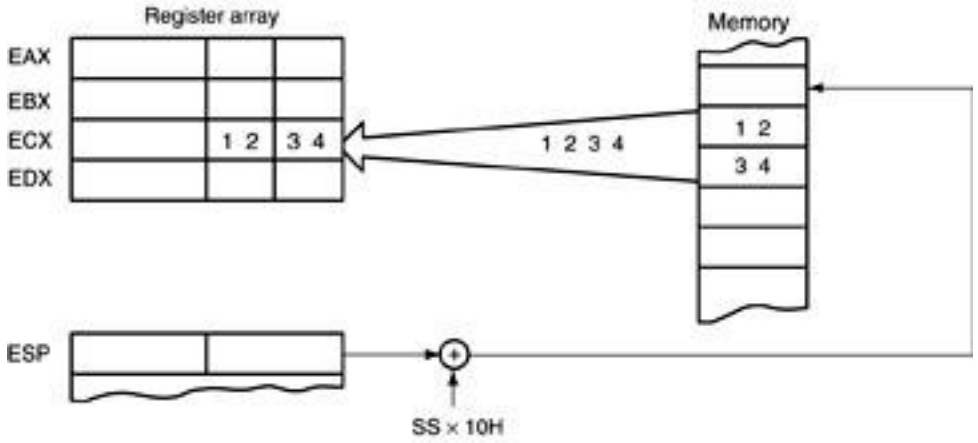
Yığın Adresleme Modları

- SP yazmacı programcının tanımladığı yığının genişliğini gösterecek şekilde ilk değer alır
- Her PUSH işleminde SP-1 ve SP-2 adreslerine 2 byte veri yazılır ve SP değeri 2 azaltılır
- Her POP işleminde SP+1 ve SP+2 adreslerinden2 byte veri okunur ve SP değeri 2 arttırılır

Yığın Adresleme Modları



(a)



(b)

SÖZDE KOMUTLAR

Sözde Komutlar

- Assembly programın düzenlemesine yöneliktir
- Doğrudan makine kodu karşılığı yoktur
- Mnemonic’ler için makine kodu karşılığı üretilir

Sözde Komutlar

- LST uzantılı dosyanın düzenlenmesi
  - PAGE, TITLE
- Program kesim düzenlemesi
  - SEGMENT/ENDS, ORG, ASSUME
- Veri tanımlamaları
  - DB, DW, DD, DQ, EQU, DUP, TYPEDEF, PTR, LABEL
- Yordam düzenleme
  - PROC/ENDP, EXTRN, PUBLIC
- MACRO düzenleme
  - MACRO/ENDM, INCLUDE, LOCAL
- Diğer
  - LENGTH, TYPE, SIZE, OFFSET, SEG, END

Sözde Komutlar

- PAGE : Derlenen .asm sonucu oluşan .lst uzantılı dosyanın satır sütün genişliğini belirler

- TITLE : Oluşan .lst için her sayfa başına yazılacak başlığı belirler

Sözde Komutlar

- SEGMENT/ENDS : Kesim tanımlaması için kullanılır

a) Hizalama :
   a) BYTE : Kesim sıradaki adresten başlar
   b) WORD : Kesim sıradaki çift adresten başlar
   c) PARA : Kesim sıradaki 16'nın tam katından başlar
   d) PAGE : Kesim sıradaki 256'nın tam katından başlar

b) Birleştirme:
   a) PUBLIC : Aynı isimli kesimlerin peşpeşe yerleşmesini sağlar
   b) COMMON : Aynı isimli kesimlerin aynı adresten başlamasını sağlar
   c) STACK : Yığın mantığında (LIFO) kesim tanımlar
   d) AT ##### : Kesim adresi belirler

Sözde Komutlar

- ORG : COM tipi programların 100H adresinden başlaması için kullanılır
- ASSUME : Kesim tanımlarının ne amaçla yapıldığı ve başlangıç adreslerini belirler

Sözde Komutlar

- DB : Define byte → 00H-0FFH arası değerleri tanımlar

  Sayi1 DB 25

  Sayi2 DB 0FFH

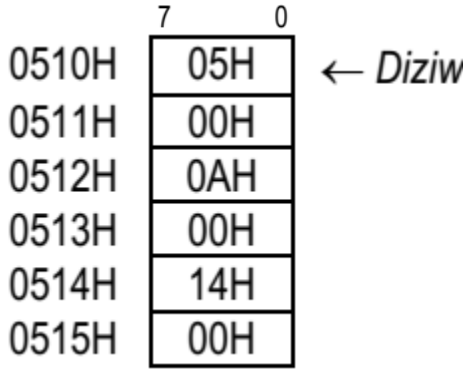
  Dizi DB 1, 2, 3, 50H

  Str DB ‘Assembly’

Sözde Komutlar

- DW : Define word → 0000H-0FFFFH arası değerleri tanımlar

  Dizi DW 5, 10, 20



- DD : Define double word → 00000000H-0FFFFFFFFH arası değerleri tanımlar

- DQ : Define quad word → 8 byte bellek alanı ayırmak için kullanılır

Sözde Komutlar

- EQU : Değişken tanımlamaz, derleme öncesi tüm EQU komutları uygun sabit değer ile değiştirilir

- DUP : Tekrarlı veri tanımı için kullanılır

```
Dizi1 DB 15 DUP (0) ; 15 adet değeri 0 olan byte ayırır
Matris DB 3 DUP (4 DUP (8)) ; 3x4 adet değeri 8 olan byte ayırır
Dizi2 DW 10 DUP (?) ; 10 adet word ayırır, ilk değer olarak o anki
                    ; bellek içeriğini kullanır
```

Sözde Komutlar

- TYPEDEF : Veri tipi tanımlama

  integer TYPEDEF WORD

  …

  i integer 9

- PTR : Bellek alanı erişim veri tipini belirler

  WORD PTR

  BYTE PTR

  FAR PTR

  NEAR PTR

Sözde Komutlar

- LABEL : Veri tanımına farklı tip ile erişim için kullanılır

```
Yenib LABEL BYTE
Sayıw DW 2535H
Yeniw LABEL WORD
Sayib DB 45H
Sayic DB 55H

MOV AX, Yeniw
MOV BL, Yenib
```

Sözde Komutlar

- PROC/ENDP
- EXTRN
- PUBLIC

Sözde Komutlar

- MACRO/ENDM
- INCLUDE
- LOCAL

Sözde Komutlar

- LENGTH : DUP ile yapılan veri tanımının boyutunu verir

  Tablo DW 15 DUP(0)

  MOV AX, LENGTH Tablo ; AX←15
- TYPE : Değişkenin (dizi ise herbir elemanıın) kaç byte yer kapladığını verir

  MOV BX, TYPE Tablo ; BX←2
- SIZE : Değişken tanımı için toplam kaç byte yer ayrıldığını verir

  MOV DX, SIZE Tablo ; DX←2x15

Sözde Komutlar

- OFFSET : Derleme öncesi değişken için ofset değeri döndürür (LEA program çalışırken ofset değeri döndürür)
- SEG : Etiketin içinde yer aldığı kesim değerini verir
- END : İlk çalıştırılacak yordamı belirler