

KARABÜK ÜNİVERSİTESİ  
TEKNOLOJİ FAKÜLTESİ MEKATRONİK MÜHENDİSLİĞİ BÖLÜMÜ




# MTM 305 MİKROİŞLEMCİLER

Arş. Gör. Emel SOYLU  
Arş. Gör. Kadriye ÖZ

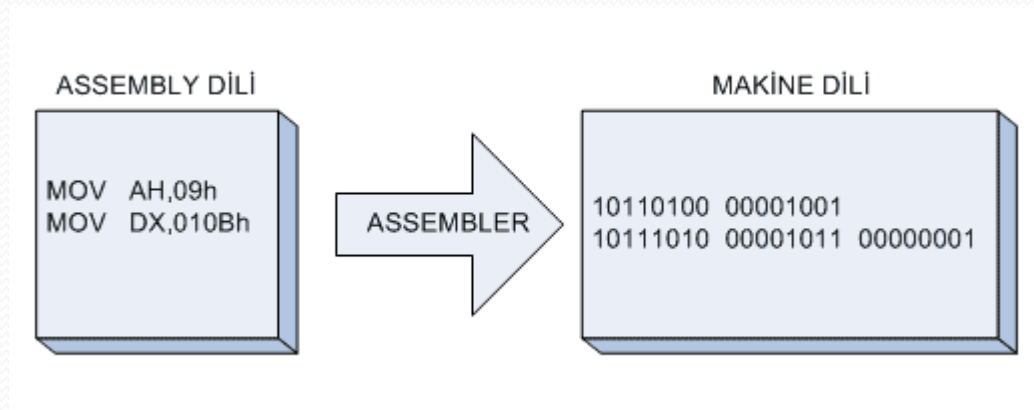



# **Assembly Dili**



Assembly programlama dili, kullanılan bilgisayar sisteminin yapısına ve işletim sistemi gibi platformlara sıkı-sıkıya bağımlı bir dildir. Assembly programlama dili düşük seviyeli bir dil olup C, C++, Pascal, C# gibi yüksek seviyeli programlama dillerine göre anlaşılması biraz daha zordur. Assembly dili ile program yazarken kullanılan bilgisayarın donanımsal özelliklerinin bilinmesi gerekir. Yazılan program kullanılan mikroişlemcinin yapısına bağlıdır. Assembly dili ile program yazarken programcı doğrudan bilgisayarın işlemcisi ve hafızası ile uğraşır. Anabellekteki (RAM'deki ) ve işlemci kaydedicilerindeki değerleri doğrudan değiştirebilme imkanı vardır.

Mikroişlemci sadece ikili sayı sisteminde yazılan komut kodlarını, başka bir ifade ile makine dilinden anlar. Assembly dilinde yazılan programları makine diline çevirmek için Assembler adı verilen çevirici(derleyici) programlar kullanılır. Aşağıda verilen şekilde Assembly dili, Makine dili ve Assembler blok olarak görülmektedir.





Bilgisayarımızda çalıştırılan tüm programlar önce bilgisayarımızın RAM belleğ'ine yüklenir. Daha sonra RAM bellekten sırası ile mikroişlemci tarafından okunarak çalıştırılır. RAM'e yüklenen veri programın makine dili karşılığından başka bir şey değildir. Yani 0 ve 1 kümeleridir.

Makine dilinde program yazmak oldukça zordur. Buna karşılık makine dili ile birebir karşılığı olan ve komutları kısaltılmış kelimelerden (mnemonik) oluşan Assembly dilinden yararlanılır.

Assembly dilinde program yazmak makine dilinde program yazmaya göre daha hızlı ve daha kolay yapılabilir. Ayrıca yazılan programların bellekte kapladıkları yerde aynıdır. Başka bir ifade ile bellek kullanımları aynıdır.

Yüksek seviyeli dillerle karşılaştırıldığında assembly dilinde yazılan programlar daha hızlıdır ve bellekte daha az yer kaplar. Buna karşılık program yazmak yüksek seviyeli dillerde daha kolaydır.

Assembly programlama dili günümüzde daha çok sistem programcıları tarafından diğer programlama dilleri içerisinde kullanılmaktadır.

## Assembly dilinin dezavantajları

- Assembly dilinde program yazmak için mikroişlemci içyapısı bilinmesi gerekir.
- Assembly dili mikroişlemci tipine göre değişir. Bir mikroişlemci için yazılan bir program başka bir mikroişlemcide çalışmayabilir. Program taşınabilir platformdan bağımsız değildir.
- Assembly dilinde program yazmak yüksek seviyeli dillere göre daha zor ve zaman alıcıdır.

## Assembly dilinin avantajları

- Bigisayar donanımı üzerinde daha iyi bir denetim sağlar. İşlemcinizin gücünü en iyi şekilde ortaya koyabilecek tek programlama dilidir.
- Küçük boyutlu bellekte az yer kaplayan programlar yazılabilir. virüslerin yazımında kullanılırlar.
- Yazılan programlar daha hızlı çalışır. Çok hızlı çalıştıkları için işletim sistemlerinde kernel ve donanım sürücülerinin programlanmasında, hız gerektiren kritik uygulamalarda kullanılmaktadır.
- Herhangi bir programlama dili altında, o dilin kodları arasında kullanılabilir.
- İyi öğrenildiğinde diğer dillerde karşılaşılan büyük problemlerin assembly ile basit çözümleri olduğu görülür.

## Assembly dilinde program yazma

Assembly dilinde program yazmak için Windows altında yer alan note pad, word pad gibi herhangi bir text editör kullanılabilir. Text editör yardımı ile Assembly dilinde program yazılır. Yazılan program TASM veya MASM assembler çevirici programları yardımı ile .obj uzantılı olarak makine diline çevrilir. Bu halde elde edilen program işletim sisteminin anladığı bir formatta değildir. TLINK bağlayıcı programı kullanılarak .exe veya .com uzantılı hale dönüştürülür. Bu haldeki program işletim sistemi üzerinde ismi yazılarak DOS ortamında çalıştırılabilir.





**Bir Assembly dilinde yazılan programda temel olarak şu bölümler bulunur:**

- Yorumlar
- Label (Etiketler)
- Talimatlar
- Komutlar

## Yorumlar / Açıklamalar

Açıklamalar program satırlarının başına noktalı virgül konularak yapılır. Açıklama satırları assembler tarafından dikkate alınmaz. Program içinde daha detaylı bilgi vermek, kullanılan komutları izah etmek için kullanılır.

örnek:

```
; MOV    ES,AX      bu komut dikkat alınmaz  
; AL ye SAYI1 değerini at
```

## Etiketler

Etiketler program içinde kullanılan özel kelimelerdir. Sonuna “:” konularak kelimenin etiket olduğu anlaşılır. Etiketlerden program akışını belirli bir noktaya yönlendirmek istediğimizde yararlanırız.

Örnek:

Son:

Basla:     JMP ANA

Burada Son, Basla kelimeleri etikettir.

# Talimatlar

## Veri tanımlama talimatları

Veri tanımlama talimatları DB, DW, DD, DF, DQ, DT ve DUP dur.

DB (Define Byte): 1 Byte'lık veri tanımlanır.

DW (Define Word): 2 Byte'lık veri tanımlanır.

DD (Define double word):: 4 Byte'lık veri tanımlanır.

DF (Define Far Word): 6 Byte'lık veri tanımlanır.

DQ (Define Quad Word): 8 Byte'lık veri tanımlanır.

DT (Define Ten Byte): 10 Byte'lık veri tanımlanır.

DUP: Duplicate

SAYI 3 DUP(0); Bellekten SAYI değişkeni için 3 byte'lık yer ayır, içeri 0 ile doldur.

SAYI DW 10 DUP(5) Bellekten SAYI değişkeni için 10x2 byte'lık yer ayır, içlerini 5 ile doldur.

## String verileri tanımlama

YAZI DB 'KARABUK'

YAZI DB 'K','A','R','A','B','U','K'

## Dizi Tanımlama

DIZI DB 2, 4, 0, -5, 7

DIZI DB 12, 0FH, 01001001B

Sayıların sonunda B olması verinin ikilik sistemde olduğunu, H olması verinin hexadesimal olduğunu gösterir. Bir şey yazılmamışsa veri onluk sistemde yazılmış anlamına gelir.


## Segment Talimatları

Segment talimatları bir segmentin başlangıcını tanımlamada kullanılır. Segmente herhangi bir isim verebilirsiniz.

SegmentAdı    SEGMENT ParametreListesi

- 
- .
- Ver tanımları ve Komutlar
- 
- 

SegmentAdı    ENDS




Parametre listesi sırası ile ALIGN, COMBINE, CLASS parametrelerini alabilir. Bu parametrelerin kullanımı seçimlidir. Bu parametreler aşağıda verilen segment tanımlamasında olduğu gibi kullanılsa da olur.

VeriSegment SEGMENT

- 
- .
- Ver tanımları ve  
Komutlar
- 
- 

VeriSegment ENDS



Parametre listesi verildiğinde aşağıdaki gibi bir tanımlama yapılabilir. Bu tanımlamada para ALIGN parametresini, public COMBINE parametresini ve 'Data' CLASS parametresini ifade eder.

VeriSegment SEGMENT para public 'Data'


- 
- .

Ver tanımları ve  
Komutlar

- 
- 

VeriSegment ENDS





**Para:** Bu alan segmentin paragraf başlarında (sonu 0 ile biten adreslerden) başlayarak yerleşeceğini ifade eder. Bu parametre belirtilmediğinde varsayılan değer para olarak belirlenir.

**Combine Alanı:** Bu alan assembler tarafından aynı adla meydana getirilen amaç programların segmentlerinin birbirleriyle nasıl bir bağ kuracağını ifade eder. Common, public, stack, memory ve at değerlerini alabilir.

**Class Alanı:** Segmentin hangi amaçla kullanılacağını ifade eder. Stack, Code ya da Data olabilir.

Örnekler :

KodSeg	SEGMENT	para public "Code"
DataSeg	SEGMENT	para public "Data"
StakSeg	SEGMENT	para Stack "Stack"

## PROC talimatı

Assembly dilinde procedure (alt program) tanımlamak için kullanılır. Altprogram aşağıda verildiği gibi tanımlanır. Far veya Near parametresi Bu alt programın aynı veya farklı segmentlerden çağrılıp çağrılmayacağını belirtir. Far olursa farklı segmentlerden, Near olursa aynı segmentten çağrılabilir. CALL AltprogAdi şeklinde çağrılarak altprogramlar kullanılır.

AltprogAdi PROC Far/Near

- .

Komutlar

- 

AltprogAdi ENDP



Beni dinlediğiniz için teşekkür ederim.