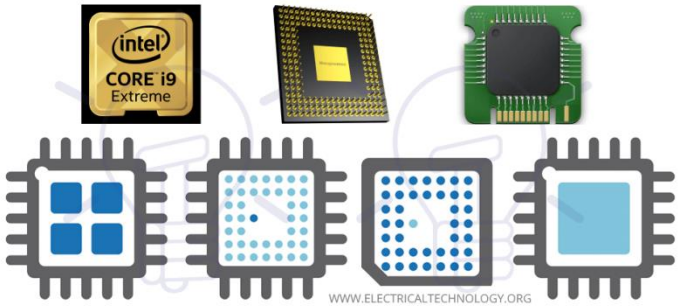


MİKROİŞLEMCİLER

Dr. Meltem KURT PEHLİVANOĞLU

W-5

MİKROİŞLEMCİLER

Digital Logic +

Digital Design +

Computer Architecture +

Microprocessors +

Microcontrollers +

Assembly Language Programming

Hatırlatma Mikroişlemciler

- Transistörlerden oluşurlar sadece 1-0 değerleri ile işlem yapabilirler,
- Transistörler birleşerek bir bitlik veriyi tutmaya yarayan flip-flopları oluşturur,
- Flip-floplar bir sonraki bit gelene kadar çıkıştaki 1 biti saklayan, clock darbesiyle birlikte çıkıştaki değeri de değiştiren lojik devredir. En basit ifadeyle 1 bit saklayabilen devredir,
- Flip-floplar birleşerek daha fazla verinin saklanmasını sağlayan registerlar oluşur,
- Bir register mikroişlemcinin yapısına göre 8,16,32 bit olabilir,
- Registerlar birleşerek işlemcinin hafıza birimlerini oluştururlar

Hatırlatma Mikroişlemciler

Transistör küçüldükçe daha çok transistörü aynı boyutlu işlemci içine sığdırabiliriz. Tüketilen enerji ve üretim maliyetleri azalır.

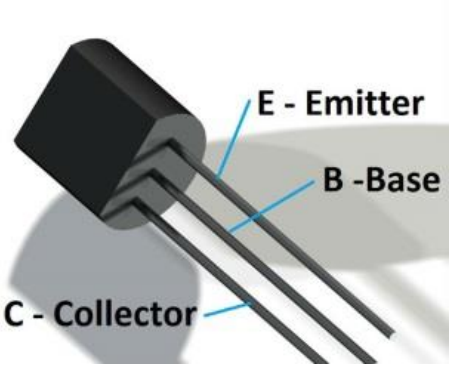
14 nanometre (nm) →

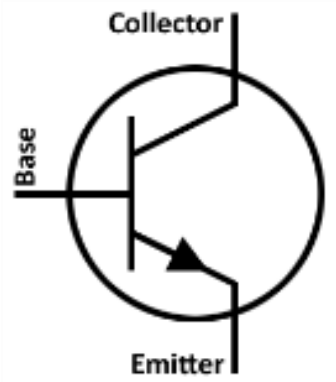
10 nm →

7nm (Apple A12 Bionic Mp iPhone XR, XS ve XS Max akıllı telefonlarında)→

5 nm(IBM)→

2,5 nm (FinFET (Fin Field-effect transistors) mikroüretim tekniği olan ‘thermal atomic layer etching’ (thermal ALE) ile 3D transistör)





Hatırlatma Mikroişlemciler

- Transistör küçüldükçe bir mikroişlemci daha fazla transistör içerebilir,
- Transistörler arası mesafe azalır,
- Transistörler daha az güç tüketir,
- Transistörler daha az ısı yayar,
- Daha yüksek performans,
- Transistör sayısı arttıkça işlemci hızı ve yapabileceği işlem sayısı artar

Hatırlatma Mikroişlemciler

Processor	MOS transistor count	Date of introduction	Designer	MOS process (nm)	Area (mm^2)
Apple A14 Bionic	11,800,000,000[127]	2020	TSMC	5 nm	***?***
HiSilicon Kirin 990 5G	10,300,000,000[117]	2019	Huawei	7 nm	113.31 mm^2
HiSilicon Kirin 990 4G	8,000,000,000[118]	2019	Huawei	7 nm	90.00 mm^2
Apple A13 (iPhone 11 Pro)	8,500,000,000[121][122]	2019	Apple	7 nm	98.48 mm^2
AMD Ryzen 7 3700X (64-bit, SIMD, caches, I/O die)	5,990,000,000[126][d]	2019	AMD	7 & 12 nm (TSMC)	199 (74+125) mm^2
AMD Ryzen 9 3900X (64-bit, SIMD, caches, I/O die)	9,890,000,000[1][2]	2019	AMD	7 & 12 nm (TSMC)	273 mm^2
AMD Epyc Rome (64-bit, SIMD, caches)	39,540,000,000[1][2]	2019	AMD	7 & 12 nm (TSMC)	1088 mm^2
AWS Graviton2 (64-bit, 64-core ARM-based, SIMD, caches)[128][129]	30,000,000,000	2019	Amazon	7 nm	***?***
Qualcomm Snapdragon 8cx / SCX8180 (octa-core ARM64 "mobile SoC", SIMD, caches)	8,500,000,000[113]	2018	Qualcomm	7 nm	112 mm^2

Hatırlatma Mikroişlemciler

- **Çekirdek** işlemci içindeki yongaya verilen isim,
- Bir nevi işlemcinin **beyni**,
- Ne kadar fazla çekirdek olursa o kadar hızlı,
- Birden fazla çekirdek aynı görevi yerine getirebilir,
- Çekirdek kendi içinde farklı parçalara ayırarak işlemleri daha hızlı yapabilir (multithreading),
- **Saat hızı ("saat oranı" veya "frekans" olarak da bilinir) (GHz)**: 1 sn'de maksimum kaç döngü (işlemciniz her saniye farklı programlardan birçok komutu (aritmetik gibi düşük seviyeli hesaplamaları) işler. Saat hızı, işlemcinizin yürüttüğü saniye başına döngü sayısını GHz (gigahertz) cinsinden ölçer) gerçekleştirebilir, her bir döngü işlemi 1 Hertz olarak kabul edilir. Programların yüklenme hızı ve ne kadar sorunsuz çalıştıkları üzerinde büyük bir etkisi vardır
- ($1Hz = 10^{-9}GHz$ veya $1GHz = 10^{9}Hz$)
- Saat hızı 3,2 GHz olan bir işlemci, saniyede 3,2 milyar döngü gerçekleştirir.

Hatırlatma Mikroişlemciler

- Bazen tek bir saat döngüsünde birden çok komut tamamlanır; diğer durumlarda, bir komut birden fazla saat döngüsünde ele alınabilir. Farklı işlemci tasarımları talimatları farklı şekilde ele aldığından, aynı işlemci markası ve nesli içindeki saat hızlarını karşılaştırmak en iyisidir.

- Örneğin, beş yıl öncesinin daha yüksek saat hızına sahip bir işlemcisi, daha düşük saat hızına sahip yeni bir işlemci tarafından geçilebilir, çünkü yeni mimari talimatlarla (instructionlarla) daha verimli bir şekilde çalışır.

- X-serisi Intel® işlemci, görevleri daha fazla çekirdek arasında böldüğü ve daha büyük bir işlemci önbelleği içerdiği için daha yüksek saat hızına sahip bir K-serisi işlemciden daha iyi performans gösterebilir.

- Ancak aynı nesil işlemciler arasında, daha yüksek saat hızına sahip olan işlemci genellikle birçok uygulamada daha düşük saat hızına sahip olandan daha iyi performans gösterir. Aynı marka ve nesilden işlemcileri karşılaştırmak bu yüzden önemlidir.

Hatırlatma Mikroişlemciler

- Mikroişlemci **önbelleği,** CPU'nun hafızadaki verilere ulaşma süresini azaltan bir donanımdır.
- Ana belleğe(RAM) kıyasla küçük, hızlı ve işlemci çekirdeğine yakındır.
- Sık kullanılan veriler ya da en güncel veriler işlemci önbelleğinde saklanır.
- Önbellek yürütücü bir program tarafından **en sık erişilen veri ve talimatları** (instruction) tutmak için kullanılır.

Hatırlatma Mikroişlemciler

Önbellek bölümleri;

- **L1 Ön Bellek (Cache):** Önemli kodlar ve veriler bellekten buraya kopyalanır. İşlemci, bu kısıma daha hızlı ulaşabilir. Kodlar için olan Code cache ve veriler için olan Data cache olmak üzere ikiye ayrılır. Kapasitesi 2 KB ile 256 KB arasında değişir.

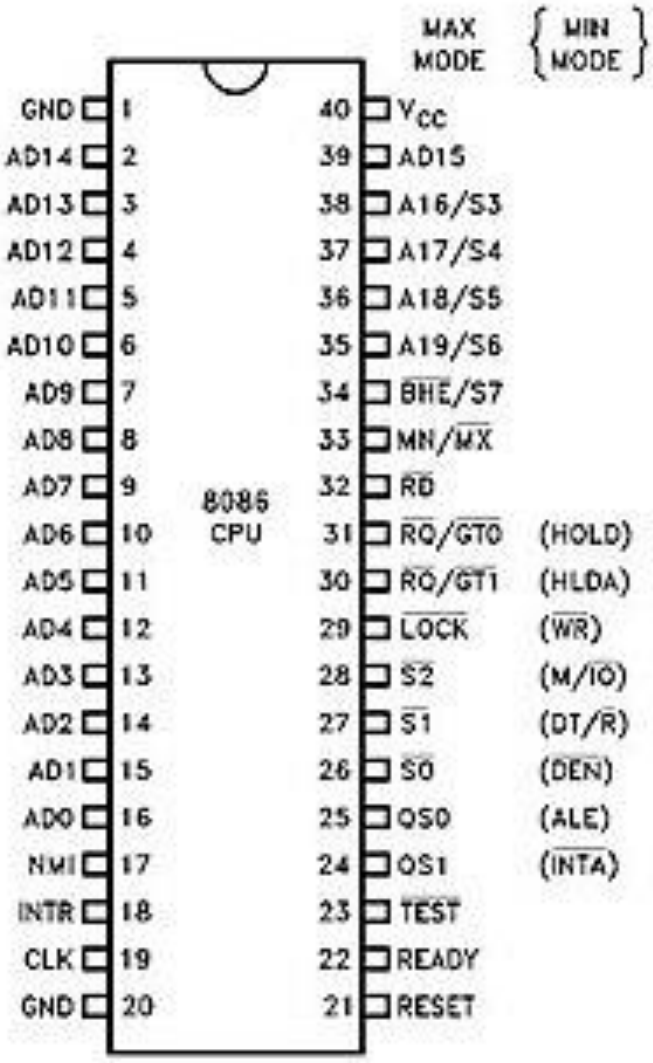
- **L2 Ön Bellek (Cache):** L1 belleklerine göre kapasiteleri, 256 KB ile 2 MB arasındadır. Başlangıçta L2 önbellek, anakart üzerinde işlemciye yakın bir yerde yer almaktaydı. Daha sonra, slot işlemciler ortaya çıkınca işlemci çekirdeğinin üzerinde kartuş şeklindeki paketlerde yer aldı. Bununla beraber çekirdeğin dışında ve işlemciyle aynı yapıda kullanılmaya başlandı. Bu kısa geçiş döneminden sonraysa L2 önbellek işlemci çekirdeklerine entegre edildi.

- **L3 Ön Bellek (Cache):** L3 ön belleklerinin kapasiteleri, 2 MB ile 256 MB arasında değişir. Çok çekirdekli işlemcilerde bütün çekirdeklere tek bir bellekle hizmet vermek akıllıca bir yaklaşım olacağı düşüncesiyle geliştirilmiştir.

  Bazen, dördüncü (L4) bir bölüm de eklenebilir.

8086 Mikroişlemcisi Adresleme Modları

Ana Kayıtçılar (register)																
AH								AL								**AX** (primary accumulator)
BH								BL								**BX** (base, accumulator)
CH								CL								**CX** (counter, accumulator)
DH								DL								**DX** (accumulator, other functions)
İndeks kayıtçıları																
SI																**S**ource **I**ndex
DI																**D**estination **I**ndex
BP																**B**ase **P**ointer
SP																**S**tack **P**ointer
Durum Kayıtçıları																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	(bit position)
-	-	-	-	O	D	I	T	S	Z	-	A	-	P	-	C	Flags
Segment kayıtçıları																
CS																**C**ode **S**egment
DS																**D**ata **S**egment
ES																**E**xtra**S**egment
SS																**S**tack **S**egment
Eğitim noktası																
IP																**I**nstruction **P**ointer



8086 Mikroişlemci

- x86 komut kümesini kullanan bütün mikroişlemciler (örneğin Intel Pentium ve AMD Athlon serisi) 8086 ile geriye dönük olarak uyumludur. 8086 işlemcisi için yazılan tüm komutlar bugünkü x86 tabanlı bilgisayarlarda çalışabilmektedir.

- Ancak günümüz bilgisayarları için yazılmış olan ve genişletilmiş x86 komut kümesi kullanan yazılımlar 8086 mikroişlemcisi üzerinde çalışamamaktadırlar

8086 Mikroişlemci

- 16 bit veri yolu; tek seferde 16 bit veri işleyebilme özelliği (16 bitlik mikroişlemci)
- 20 bitlik adresleme özelliği; 2^20 = 1MBlık belleğe erişebilme
- 8 adet genel amaçlı 16 bitlik register

- **AX** - accumulator register – akümülatör (**AH / AL**).
- **BX** - the base address register – adres başlangıcı (**BH / BL**).
- **CX** - the count register – sayma (**CH / CL**).
- **DX** - the data register – veri (**DH / DL**).
- **SI** - source index register – kaynak indisi.
- **DI** - destination index register – hedef indisi.
- **BP** - base pointer – temel gösterici.
- **SP** - stack pointer – yığıt gösterici.

8086 Mikroişlemci

- Segment registerlarının özel amaçları vardır, bellekte ulaşılabilir bazı bölümleri işaretler.
- Segment registerları, genel amaçlı registerları ile birlikte çalışarak hafızada herhangi bir bölgeyi işaretleyebilir.

- **CS** – (Code Segment) Mevcut programın bulunduğu bölümü işaretler.
- **DS** – (Data Segment) Genellikle programda bulunan değişkenlerin bulunduğu bölümü işaretler.
- **ES** – (Extra Segment) Bu register'ın kullanımı, kullanıcıya bırakılmıştır.
- **SS** – (Stack Segment) yığının bulunduğu bölümü işaretler.

8086 Mikroişlemci

- Örneğin;

AX= 0011000000111001b ise AH=00110000b ve AL=00111001b olur

- AX ← 1234H
- BX ← ABCDH
- MOV AX, BX
- AX ← ABCDH
- BX ← ABCDH

MOV DST, SRC

- DST ← SRC

8086 Mikroişlemci

- 8086, **20 bitlik adres hattı**; (2^20=1048576=1MB) **1MBlık belleği** adresleyebilir.

- Ancak 16-bitlik registerlar ile en fazla adreslenebilir bellek uzayı 64 KB büyüklüğündedir. Çünkü tüm dahili registerlar 16-bit büyüklüğündedir (2^16=65536=64KB (örn. 0000h-FFFFh))

- 64 KB'lık sınırların dışında programlama yapabilmek için extra operasyonlar (adres bölütleme-segmentation) kullanmak gerekir

8086 Mikroişlemci

- 64 KB'lık erişilebilecek adres uzayı bölütlü (segment) adresleme ile 1MB a (00000h-FFFFFh) çıkarılır;
- Şöyle ki

Fiziksel adres = F A = bölüt (segment) × 16 + offset

Böylece 20 bitlik fiziksel bellek adresi hem adres registerındaki offset, hem de segment registerındaki paragraf adresi değerinden hesaplanır.

(İpucu: program kodu için kod bölütü CS, veri adresleme için veri bölütü DS, extra bölüt ES, yığın adresleme için SS)

8086 Mikroişlemci

- Örneğin, fiziksel adres 12345h (heksadesimal) işaretlenmesi isteniyor ise, DS = 1230h ve SI = 0045h olmalıdır.

- CPU, segment register'ı 10h ile çarpar (çünkü fiziksel adres hexadecimal- decimal olsaydı 16 ile çarpacaktı) ve genel amaçlı register'da bulunan değeri de ilave eder (1230h×10h + 45h = 12345h).

(**DS:Data Segment, SI: Source Index**)

8086 Mikroişlemci

- **2 register** tarafından oluşturulmuş olan **adrese**, effective address (**efektif adres**) ismi verilir

- BX, SI (Source Index) ve DI (Destination Index) register'ları, DS (Data Segment) ile birlikte çalışır;
- BP (Base Pointer) ve SP (Stack Pointer)register'ları ise SS (Stack Segment) ile birlikte çalışır.

- Diğer genel amaçlı registerlar, efektif adres oluşturmak için kullanılmazlar. Ayrıca, BX efektif adres oluşturulmasında kullanılırken, BH ve BL kullanılmaz.

8086 Mikroişlemci

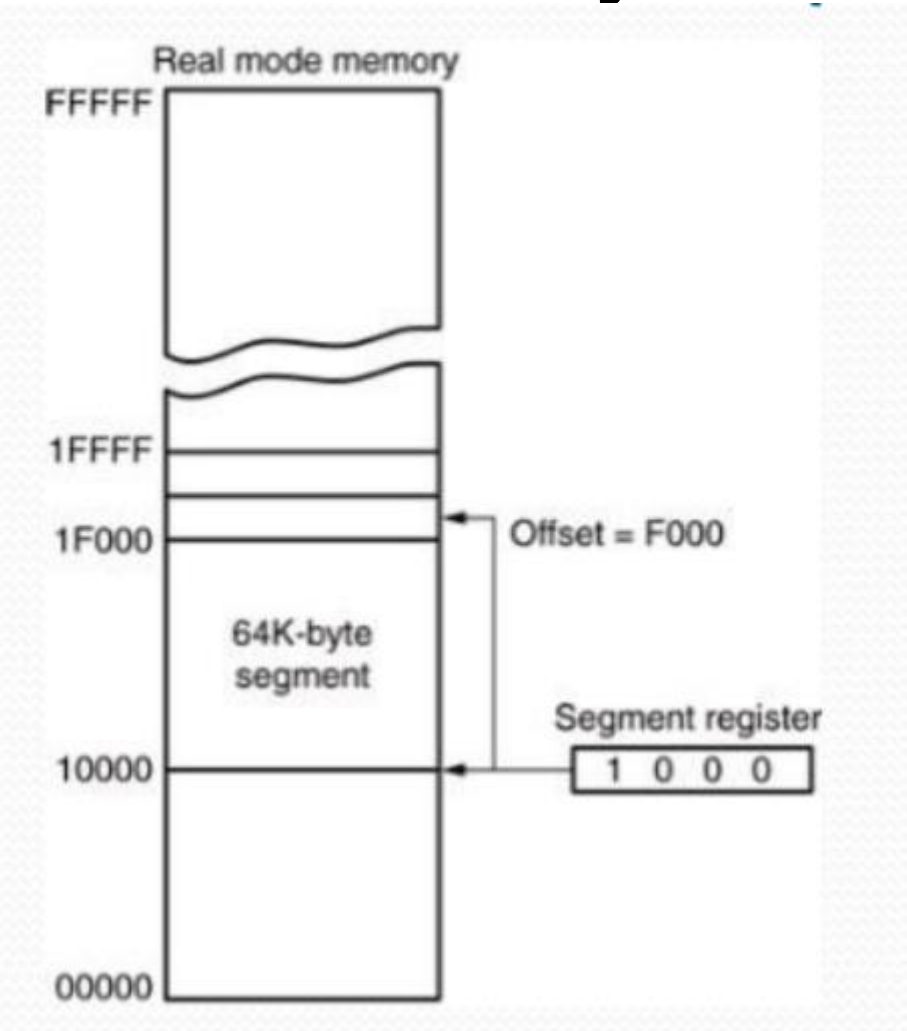
- Tüm fiziksel bellek adresleri

segment adresi+ **offset adresi** ilave edilmesi ile bulunur.

  - **segment adresi:** Herhangi bir 64 KB'lık hafıza bölümünün başlangıcını gösterir.

  - **offset adresi:** 64 KB'lık hafıza bölümünde herhangi bir satırı belirtir.

8086 Mikroişlemci



- Segment register'ı 1000h değerine sahip ise, 10000h (CPU, segment register'ı 10h ile çarpar segment adresi bulur) ile 1FFFFh aralığında bir hafıza adresine karşılık gelir.
    - 64KB'lık bir aralıktır
- Offset değeri F000h ise 1F000h adresindeki hafıza satırına karşılık gelir (segment adresi+offset adresi 10000h+F000h=1F000h)

CPU, segment register'ı 10h ile çarpar

8086 Mikroişlemci

- Başlangıç adresi belli ise, bitiş adresi FFFFh ilave edilerek bulunur (16-bitlik segment register'ı 64 KB'lık bir bölümü işaretler)
- Offset adresi, segment adresine ilave edilir ve bellek adresi bulunur.
- Segment adresi ve offset adresi ' **1000:2000** ' biçiminde de yazılabilir.
  - Bu durumda segment adresi 1000h ve offset'te 2000h'dir.

8086 Mikroişlemci

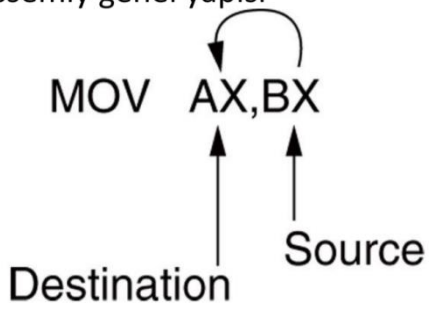
- CS(Code Segment) kod bölümünün başlangıcına işaret eder. IP(Instruction Pointer) kod bölümü içerisinde bir sonraki komutun bulunduğu bellek adresine işaret eder.
- Eğer **CS=1400h** ve **IP=1200h**, mikroişlemci, bir sonraki komutu
  - 14000h (1400hx10h=14000h) + 1200h = 15200h adresinden okur.

8086 Mikroişlemci Adresleme Modları

- Adresleme modları belleğin nasıl kullanıldığını, belleğe nasıl erişileceğini ve verilerin belleğe nasıl yerleştirileceğini belirler.

- REG: AX, BX, CX, DX, AH, AL, BL, BH, CH, CL, DH, DL, DI, SI, BP, SP.
- SREG: DS, ES, SS, and only as second operand: CS.
- bellek: [BX], [BX+SI+7] ...
- immediate: 5, -24, 3Fh, 10001101b ...

- 8086 assemly genel yapısı



8086 Mikroişlemci

Hemen Adresleme (Immediate Addressing)

- **Sabit bir değer** (immediate) bir registera aktarılır.
- Sabit değerin büyüklüğü ile register uyumlu olmalıdır.
- Örneğin 8 bitlik bir registera 16 bitlik bir değer yüklenemez.

- Genel Kulanımı: **KOMUT register, immediate**

  MOV CL, 16h
  MOV DI, 2ABFh
  MOV AL, 4567h (Yanlış kullanım-8 bitlik register AL)

8086 Mikroişlemci

Doğrudan adresleme (Direct addressing)

- Doğrudan bir adres değeri kullanılır diğer bir ifadeyle erişilecek hafıza gözünün doğrudan gösterildiği durumdur.

- Bir adresten bir registera veri aktarımı gerçekleştirilir. Bir başka ifade ile operandlardan birisi adres belirtir.
- Genel kullanımı:
  - **KOMUT register, memory veya**
  - **KOMUT memory, register**
- Örnek:

```
MOV AX, [1000h] ; 1000h adresindeki değeri AX e ata

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

8086 Mikroişlemci

Kaydedici Adresleme (Register Addressing)

- Bu adresleme modunda her iki operand da registerdır.

- Genel kullanımı: **KOMUT register, register**

Örnek:

MOV AL, BL

INC BX

DEC AL

SUB DX, CX

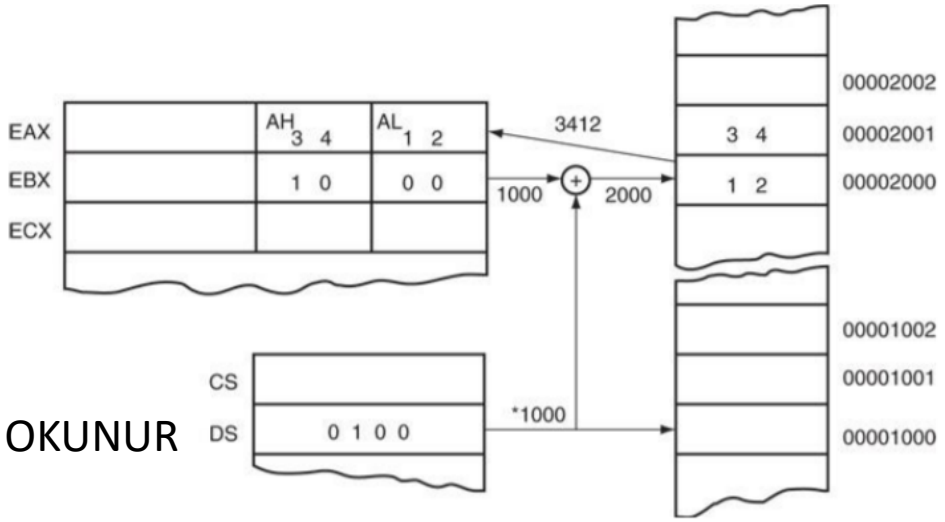
8086 Mikroişlemci

Register Dolaylı adresleme (Register Indirect addressing)

- Etkin adres değeri (**offset**) BX, BP, SI, DI registerlarından birinde bulunur.
- BP (SS ile), BX, DI ve SI (DS ile) yazmaçları ile kullanılabilir
- Genel kullanımı:
  - **KOMUT register, [BX/BP/SI/DI] veya**
  - **KOMUT [BX/BP/SI/DI], register**

- Örnek:

MOV AX, [BX] ; AX ←**DS:BX**

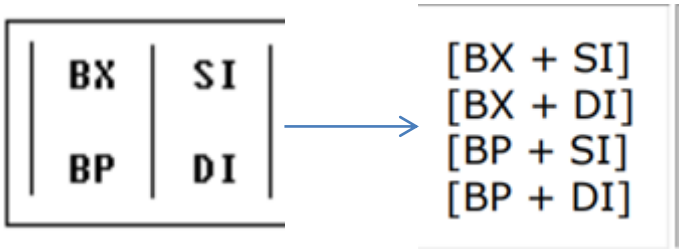


SEGMENT ADRES:OFFSET ADRES

Komut SEGMENT ADRES*10+OFFSET ADRESTEN OKUNUR

8086 Mikroişlemci

BASE+INDEX Adresleme



- Temelde bir dolaylı adresleme modudur
- Base registerı (BX veya BP) işlem yapılacak bellek konumunun başlangıcını göstermek için kullanılır
- Index registerları (DI veya SI) verinin bu başlangıç adresine görece yerini tutmak için kullanılır
- Genel kullanımı:
  - **KOMUT register, [BX+SI] / [BX+DI] / [BP+SI] / [BP+DI] veya**
  - **KOMUT [BX+SI] / [BX+DI] / [BP+SI] / [BP+DI], register**

    MOV DX, [BX+DI]

BX ve BP asla toplanmaz!

8086 Mikroişlemci

[SI + d8]	[SI + d16]
[DI + d8]	[DI + d16]
[BP + d8]	[BP + d16]
[BX + d8]	[BX + d16]

Yazmaç Göreli Adresleme (Register Relative Addressing)

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

Sabit offset değerleri:d8 veya d16

d8 - 8 bit işaretli immediate (örneğin: 22, 55h, -1,...)

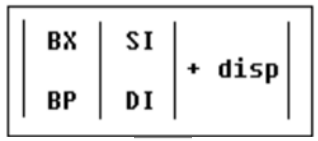
d16 -16 bit işaretli immediate (örneğin: 300, 5517h, -259,...).

- Genel kullanımı:
  - **KOMUT register, [BX+d8] / [DI+d8] / [BP+d8] / [SI+d8] veya**
  - **KOMUT [BX+d8] / [DI+d8] / [BP+d8] / [SI+d8], register**

  - **KOMUT register, [BX+d16] / [DI+d16] / [BP+d16] / [SI+d16] veya**
  - **KOMUT [BX+d16] / [DI+d16] / [BP+d16] / [SI+d16] , register**

8086 Mikroişlemci

Base Relative + Index Addressing



[BX + SI + d16]
[BX + DI + d16]
[BP + SI + d16]
[BP + DI + d16]

[BX + SI + d8]
[BX + DI + d8]
[BP + SI + d8]
[BP + DI + d8]

- Base index ve yazmaç göreceli adresleme modlarının birleşimi gibi düşünebiliriz.

- Genel kullanımı:
  - **KOMUT register, [BX+SI+d8] / [BX+DI+d8] / [BP+SI+d8] / [BP+DI+d8] / [BX+SI+d16] / [BX+DI+d16] / [BP+SI+d16] / [BP+DI+d16] veya**

  - **KOMUT [BX+SI+d16] / [BX+DI+d16] / [BP+SI+d16] / [BP+DI+d16] / [BX+SI+d8] / [BX+DI+d8] / [BP+SI+d8] / [BP+DI+d8], register**

MOV AX, [BX + SI] + 25 ile MOV AX, [BX + SI+25] aynıdır

8086 Mikroişlemci
Base Relative + Index Addressing

- Örneğin;

DS = 100,

BX = 30,

SI = 70

Mikroişlemci tarafından hesaplanan fiziksel adres [BX + SI] + 25

= 100 * 16 + 30 + 70 + 25 = 1725 olur

MOV AX, [BX + SI] + 25 ; 1725 fiziksel adresindeki değeri AX registerına at

8086 Mikroişlemci

- KOMUT memory, memory

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

- **Sabit bir değer** doğrudan **Segment Registerine atanamaz**

MOV DS,1234 Yanlıştır.

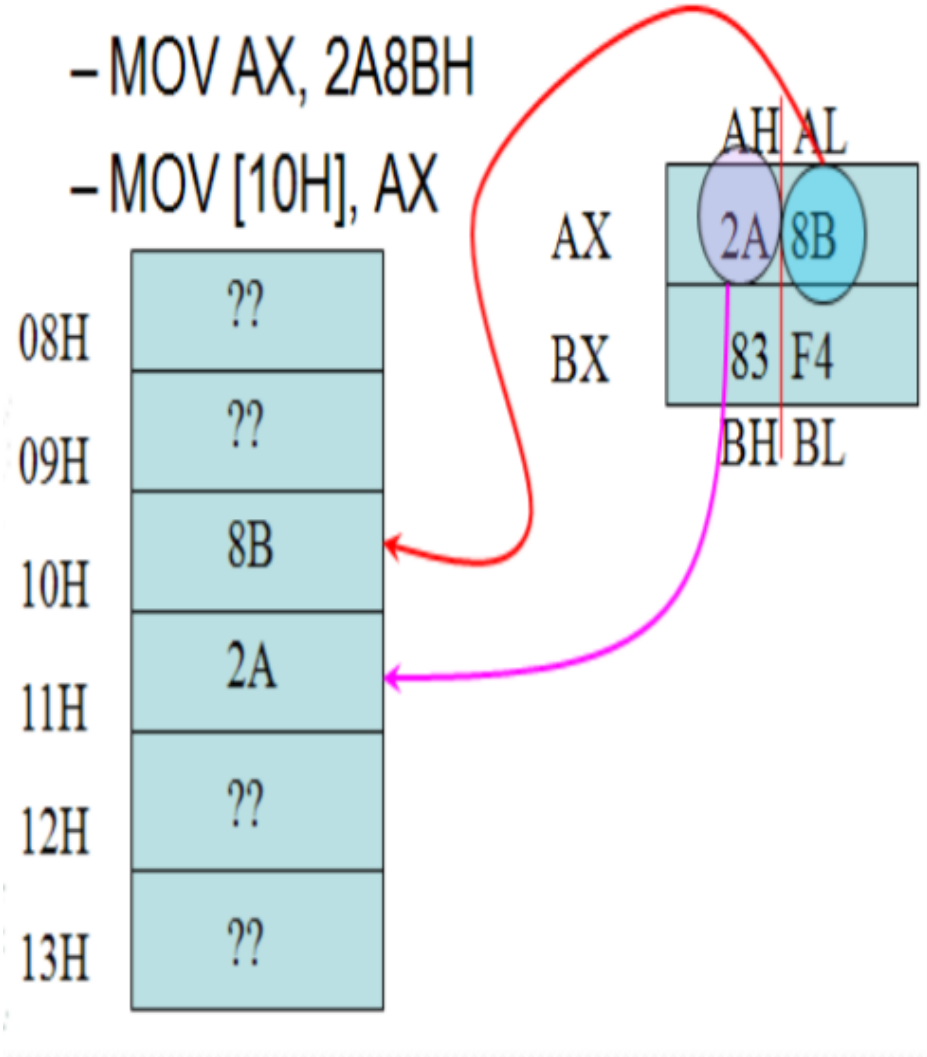
Bunun yerine aşağıdaki kullanım geçerlidir.

MOV AX, 1234 ; önce registera

MOV DS, AX ; sonra segment kaydedicisine değer atanır.

8086 Mikroişlemci
16 Bitlik 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.
- Yanda verilen örnekte AX kaydedicisinde 2A8BH değeri yer almaktadır.
- Düşük değerlikli değer AL de yer almakta ve 10H adresine yerleşmektedir. (alt byte yani AL ilk)
- Yüksek değerlikli kısımda yer alan AH daki değer 2AH değeri ise 11H adresine yerleşmektedir (Üst byte (AH) ise bir sonraki adreste tutulur)



8086 Mikroişlemci
Yığın Adresleme

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

PUSH CS ; çalışır

POP CS ; assembler hatası verir

- 8086'da yığın geçici veri saklamak için ve fonksiyonlardan dönüşlerde dönüş adreslerini saklamak için kullanılır

8086 Mikroişlemci
Yığın Adresleme

- Yığın LIFO mantığında çalışır (Last In First Out)
- Yığın ile ilgili PUSH ve POP komutları kullanılır.
- 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 adreslerinden 2 byte veri okunur ve SP değeri 2 arttırılır

8086 Mikroişlemci
Yığın Adresleme

- MOV AX, 0200H AX yazmacına 0200H verisi yazılmakta
- MOV SS, AX Bu saklanan veri SS yazmacına yüklenmektedir
- MOV SP, 1238H SP'ye 1238H verisi yüklenmektedir
- MOV AX, 0AFFH AX=0AFFH
- MOV BH, 11H BH=11H
- MOV BL, 12H BL=12H
- MOV CX, 01H CX=01H
- PUSH AX PUSH AX komutu ile AX yazmacının değeri 0200:1238 ve 0200:1236 adresleri arasına yazılır. AX yazmacının boyutunun 2 bayt olması nedeniyle bu saklama işleminin ardından SP yazmacının değeri 2 azalır.
- PUSH BX BX yazmacının iki bayt uzunluğundaki değeri 0200:1236 ve 0200:1234 adreslerinin arasına yazılır.
- POP AX SP'nin gösterdiği adresten başlayarak iki bayt okur ve AX yazmacına yazar. AX yazmacının değeri 1112H olarak belirlenmiş olur
- PUSH CX 0200:1236 ve 0200:1234 adreslerinin arasına CX'in değeri olan 0001H yazılır.
- POP BX SP yazmacı ile gösterilen yığıtın tepesinden iki baytlık bir veri çeker ve bu veriyi BX yazmacına yazar. Bu işlemin ardından 0200:1236 adresi ile 0200:1234 adresi arasında kalan verileri BX'e aktarılırken SP imleci de iki artırılarak yığıtın yeni tepe noktası olan 0200:1236 adresini göstermesi sağlanır
- POP CX CX yazmacına 0200:1236 ve 0200:1238 adreslerinin arasındaki adreslerde bulunan verileri CX yazmacına atar.

8086 Mikroişlemci
Yığın Adresleme

1	MOV AX, 0200H
2	MOV SS, AX
3	MOV SP, 1238H
4	MOV AX, 0AFFH
5	MOV BH, 11H
6	MOV BL, 12H
7	MOV CX, 01H
8	PUSH AX
9	PUSH BX
10	POP AX
11	PUSH CX
12	POP BX
13	POP CX

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 adreslerinden 2 byte veri okunur ve SP değeri 2 arttırılır

8086 Mikroişlemci
Yığın Adresleme

0200:1232	
0200:1233	
0200:1234	
0200:1235	
0200:1236	
0200:1237	
0200:1238	
0200:1239	
0200:1249	

1. Başlangıç durumu

0200:1232	
0200:1233	
0200:1234	
0200:1235	
0200:1236	**FF**
0200:1237	**0A**
0200:1238	
0200:1239	
0200:1249	

2. PUSH AX komutundan sonra

0200:1232	
0200:1233	
0200:1234	**12**
0200:1235	**11**
0200:1236	**FF**
0200:1237	**0A**
0200:1238	
0200:1239	
0200:1249	

3. PUSH BX komutundan sonra

0200:1232	
0200:1233	
0200:1234	
0200:1235	
0200:1236	**FF**
0200:1237	**0A**
0200:1238	
0200:1239	
0200:1249	

4. POP AX komutundan sonra

0200:1232	
0200:1233	
0200:1234	**01**
0200:1235	**00**
0200:1236	**FF**
0200:1237	**0A**
0200:1238	
0200:1239	
0200:1249	

5. PUSH CX komutundan sonra

0200:1232	
0200:1233	
0200:1234	
0200:1235	
0200:1236	**FF**
0200:1237	**0A**
0200:1238	
0200:1239	
0200:1249	

6. POP BX komutundan sonra

0200:1232	
0200:1233	
0200:1234	
0200:1235	
0200:1236	
0200:1237	
0200:1238	
0200:1239	
0200:1249	

7. POP CX komutundan sonra

Aşağıda sırasıyla AX, BX ve CX yazmaçlarının programdan önceki ve sonraki durumları gösterilmiştir.

	Önce	**Sonra**
AX	0AFF	1112
BX	1112	0001
CX	0001	0AFF

1	MOV AX, 0200H
2	MOV SS, AX
3	MOV SP, 1238H
4	MOV AX, 0AFFH
5	MOV BH, 11H
6	MOV BL, 12H
7	MOV CX, 01H
8	PUSH AX
9	PUSH BX
10	POP AX
11	PUSH CX
12	POP BX
13	POP CX

8086 Mikroişlemci
Ayrılmış Bölgeler

- Belleğin bazı bölgeleri, bir takım özel verileri tutmak için ayrılmıştır.

- Örneğin belleğin 0000H ile 03FFH adresleri arasında kalan bellek alanı kesme yöneyleri (vektörleri) için ayrılmıştır.

- FFFF0H ya da FFFFFH adresleri bilgisayar ilk açıldığında ya da yeniden başlatıldığında başlangıç adresi olarak kullanılır.

8086 Mikroişlemci
Kesmeler

- Kesmeler, işlemcinin içinde çalışan programın akışını durdurup denetimi özel bir kesme yordamına aktaran özel komutlar ya da sinyallerdir.

- Çoğu kesme komutunun işlenmesinin ardından kesme yordamı işini tamamladığında program kaldığı yerden çalışmayı sürdürür.

- Kesmeler yazılım ya da donanım kullanılarak uygulanabilir.

8086 Mikroişlemci Komutlar

8086 mikroişlemcisinin komut kümesi şu alt başlıklara ayrılabilir:

Veri Taşıma Komutları: MOV, PUSH, POP, XCHG, IN, OUT, XLAT, XLATB, LEA, LDS, LES, LAHF, SAHF, PUSHF, POPF

• Aritmetik İşlemi Komutları: ADD, ADC, INC, AAA, DAA, SUB, SBB, DEC, NEG, CMP, AAS, DAS, MUL, IMUL, AAM, DIV, IDIV, AAD, CBW, CWD

• Mantık İşlemi Komutları: NOT, SHL, SAL, SHR, SAR, ROL, ROR, RCL, RCR, AND, TEST, OR, XOR

• Dizgi İşleme Komutları: REP / REPE / REPNE / REPNZ / REPZ, MOVSB / MOVSW, CMPSB / CMPSW, SCASB / SCASW, LODSB / LODSW, STOSB / STOSW

• Denetimi Aktarma Komutları: CALL, JMP, RET / RETF, JE / JZ, JL / JNGE, JLE / JNG, JB / JNAE, JBE / JNA, JP / JPE, JO, JS, JNE / JNZ, JNL / JGE, JNLE / JG, JNB / JAE, JNBE / JA, JNP / JPO, JNO, JNS, LOOP, LOOPZ / LOOPE, LOOPNZ / LOOPNE, JCXZ, INT, INTO, IRET

• İşlemci Denetimi Komutları: CLC, CMC, STC, CLD, STD, CLI, STI, HLT, WAIT, ESC, LOCK, NOP

8086 Mikroişlemci Komutlar

- Yazılan bir programın çalıştırılabilmesi için programı oluşturan komutların bir yerden çekilmesi ve çalıştırılması gerekir.
- 8086 işlemcisi program kodunu CS yazmacı ile gösterilen kod bölütünden (code segment) çeker ve çalıştırır.
- Kod bölütünden bir komutun okunması için komutun bellekte bulunduğu gerçek adresin bilinmesi gerekir.
- Gerçek adres, bölüt adresi ve göreceli konum adreslerinden hesaplanabildiğinden, bir komut, belleğin bir bölütünden çekilirken bu bölütün neresinden çekildiğini belirtmek amacıyla göreceli konum adresinin de bölüt adresi ile birlikte belirtilmesi gerekir.
- Bu yüzden bir komutun bellekten getirilmesi için CS'nin yanında IP (instruction pointer – komut imleci) adresi de verilmektedir.

Örneğin 0200:0000 adresindeki komut çekilip çalıştırıldıktan sonra 0200:0001 adresine geçilir.
0200:0001 sanal adresine denk gelen gerçek adres:
0200H x 10H = 02000H →02000H + 0001H = 02001H şeklinde hesaplanarak 02001H olarak bulunur.

8086 Mikroişlemci Komutlar

CS:IP	İşlem Komutu ve İşlenen	Çevirici Komutları
1234:0000	B3 53	MOV BL, 'S'
1234:0002	B4 43	MOV AH, 'C'
1234:0004	B9 12 00	MOV CX, 12H
1234:0007	BA 13 00	MOV DX, 13H
1234:000A	03 CA	ADD CX, DX

Dec	Hx	Oct	Html	Chr
64	40	100	@	@
65	41	101	A	A
66	42	102	B	B
67	43	103	C	C
68	44	104	D	D
69	45	105	E	E
70	46	106	F	F
71	47	107	G	G
72	48	110	H	H
73	49	111	I	I
74	4A	112	J	J
75	4B	113	K	K
76	4C	114	L	L
77	4D	115	M	M
78	4E	116	N	N
79	4F	117	O	O
80	50	120	P	P
81	51	121	Q	Q
82	52	122	R	R
83	53	123	S	S

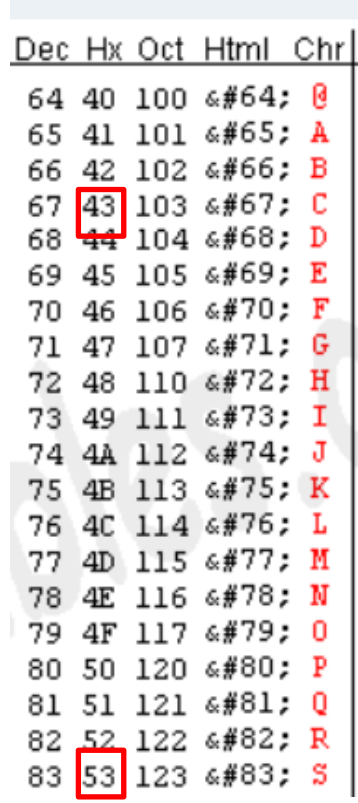
İşlemci, komutları işlemeye

1. 1234:0000 adresinden başladığında 1234:0000 adresindeki B3H ile 1234:0001 adresindeki 53H verisini yakalar ve bu verilerin MOV BL, 'S' çevirici dil koduna denk geldiğini algılar. Bir başka deyişle, MOV BL, 'S' çevirici kodu bir çevirici ile çevrilirse onaltılık tabandaki B3 ve 53 baytlarına dönüşecektir. Çevrilmiş bu çevirici dil kodu 1234:0000 ve 1234:0001 adreslerinde bulunmaktadır.

1234:0000'da MOV komutunun değişmez/sabit bir verinin değerini (S harfinin ASCII değerini) BL yazmacına aktaran bir çeşidi olan B3H ve 1234:0001'de bu değişmez/sabit verinin değeri (S harfinin onaltılık tabandaki ASCII değeri) bulunmaktadır. B3H 53H işlendikten sonra 1234:0002 adresine geçilir.

8086 Mikroişlemci Komutlar

CS:IP	İşlem Komutu ve İşlenen	Çevirici Komutları
1234:0000	B3 53	MOV BL, 'S'
1234:0002	B4 43	MOV AH, 'C'
1234:0004	B9 12 00	MOV CX, 12H
1234:0007	BA 13 00	MOV DX, 13H
1234:000A	03 CA	ADD CX, DX

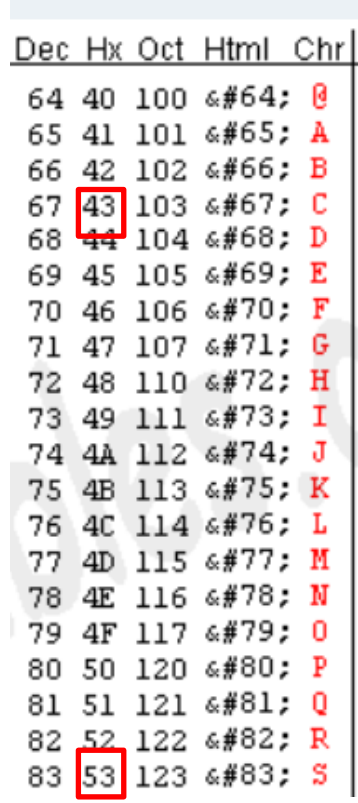


Dec	Hx	Oct	Html	Chr
64	40	100	@	@
65	41	101	A	A
66	42	102	B	B
67	43	103	C	C
68	44	104	D	D
69	45	105	E	E
70	46	106	F	F
71	47	107	G	G
72	48	110	H	H
73	49	111	I	I
74	4A	112	J	J
75	4B	113	K	K
76	4C	114	L	L
77	4D	115	M	M
78	4E	116	N	N
79	4F	117	O	O
80	50	120	P	P
81	51	121	Q	Q
82	52	122	R	R
83	53	123	S	S

2. 1234:0002 adresinde AH yazmacına C harfinin onaltılık tabandaki ASCII değerini aktaran B4 işlem komutu ve 1234:0003 adresinde C harfinin onaltılık tabandaki değeri olan 43H bulunmaktadır.
   - MOV komutu BL yazmacına değişmez bir veri aktarırken B3H işlem komutuna çevrilirken, AH yazmacına değişmez bir veri aktarılırken B4H işlem komutuna çevrilmektedir. Bunun nedeni MOV komutunun yalnızca bir tane işlem komutu bulunması durumunda bu işlem komutu ile yalnızca tek bir yazmaca veri aktarabilmesidir. B3H veya B4H gibi işlem komutlarının nasıl hesaplandığı çevirici dilinin komutları detaylı olarak açıklanırken belirtilecektir. Bu aşamada tek bilinmesi gereken bir komutun farklı işlem biçimleri olduğunda bu komutun çevrilmesinden sonra ortaya çıkan işlem komutlarının değişiklik göstereceğidir.

8086 Mikroişlemci Komutlar

CS:IP	İşlem Komutu ve İşlenen	Çevirici Komutları
1234:0000	B3 53	MOV BL, 'S'
1234:0002	B4 43	MOV AH, 'C'
1234:0004	B9 12 00	MOV CX, 12H
1234:0007	BA 13 00	MOV DX, 13H
1234:000A	03 CA	ADD CX, DX



Dec	Hx	Oct	Html	Chr
64	40	100	@	@
65	41	101	A	A
66	42	102	B	B
67	43	103	C	C
68	44	104	D	D
69	45	105	E	E
70	46	106	F	F
71	47	107	G	G
72	48	110	H	H
73	49	111	I	I
74	4A	112	J	J
75	4B	113	K	K
76	4C	114	L	L
77	4D	115	M	M
78	4E	116	N	N
79	4F	117	O	O
80	50	120	P	P
81	51	121	Q	Q
82	52	122	R	R
83	53	123	S	S

3. 1234:0004 adresindeki veri okunduğunda MOV CX, 12H görülecektir. Bu kod bir işlem komutu ve iki tane işlenen olmak üzere 3 baytlık alan kaplamaktadır. MOV CX, 12H kodu 12H baytını CX yazmacına kaydeden bir koddur. MOV BL, 'S' 2 baytlık yer kaplarken, MOV CX,12H 3 baytlık yer kaplamasının nedeni BL yazmacının 8 bitlik olmasıdır. MOV CX, 12H kodu CX yazmacına bir baytlık 12H verisini, CX yazmacının alt yarısına (CL) 12H ve üst yarısına (CH) 00H olarak yazar. Böylece MOV komutu CX yazmacının üzerine uygulanarak hem CL hem de CH yazmaçları değiştirilmiş olur. CX yazmacının boyutunun toplam iki bayt olması nedeniyle B9 ile kodlanan MOV komutu iki baytlık işlenen kullanmakta ve işlem komutuyla birlikte MOV komutu toplam üç bayt yer tutmaktadır. Burada dikkat edilmesi gereken nokta CX yazmacına aktarılan 0012H değerinin işlenen olarak 12 00 sırasıyla MOV komutuna verilmesidir. Bunun nedeni Intel mikroişlemcilerinin çoğunda (ve 8086 işlemcisinde) "Küçüğü Başta (Little Endian)" Kuralı'nın kullanılmasıdır.

8086 Mikroişlemci Komutlar

CS:IP	İşlem Komutu ve İşlenen	Çevirici Komutları
1234:0000	B3 53	MOV BL, 'S'
1234:0002	B4 43	MOV AH, 'C'
1234:0004	B9 12 00	MOV CX, 12H
1234:0007	BA 13 00	MOV DX, 13H
1234:000A	03 CA	ADD CX, DX

Örnekte 1234:0007, 1234:0008 ve 1234:0009 adreslerindeki BA 13 00 baytları MOV DX,13H komutunu oluşturmaktadır. Önceki MOV CX,12H komutunda olduğu gibi ilk bayt olan BA, DX yazmacına veri aktarılacağını ardından gelen iki bayt ise aktarılacak olan değerin 13H olduğunu göstermektedir.

1234:000A ve 1234:000B adreslerinde ise ADD CX,DX kodu bulunmaktadır. Bu kod CX ve DX yazmaçlarının değerlerini toplayıp sonucu CX yazmacına yazar ve bellekte saklanan 03 CA baytları bu komutun çevrilmiş biçimidir.

Kaynaklar

- https://www.electricaltechnology.org/2020/05/types-of-microprocessors.html

- http://www.lojikprob.com/elektronik/mikroislemci-mimarisi-2-adres-veri-ve-kontrol-yollari/
- https://www.academia.edu/11730974/KTU_M%C4%B0KRO%C4%B0%C5%9ELEMC%C4%B0LER_DERS_NOTLARI
- https://tr.wikipedia.org/wiki/Intel_8086#:~:text=8086%20(ayr%C4%B1ca%20iAPX86%20de%20denir,aylar%C4%B1nda%20ilk%20%C3%A7ip%20piyasaya%20s%C3%BCr%C3%BClm%C3%BC%C5%9Ft%C3%BCr.
- https://www.tutorialspoint.com/microprocessor/microprocessor_8086_overview.htm
- http://rakeshharsha.blogspot.com/2017/05/the-8086-microprocessor-architecture.html
- https://tr.wikipedia.org/wiki/X86
- https://electronicsdesk.com/8086-microprocessor.html
- https://www.intel.com.tr/content/www/tr/tr/gaming/resources/cpu-clock-speed.html
- https://tr.wikipedia.org/wiki/%C4%B0%C5%9Flemci_%C3%B6nbelle%C4%9Fi
- https://www.karel.com.tr/blog/ram-ve-onbellek-arasindaki-fark-nedir
- http://www.scozturk.com/wp-content/uploads/permanent/scozturk.com%20-%20Intel%208086%20ile%20Mikroi%C5%9Flemci%20Programlamaya%20Giri%C5%9F%20-%20Kitap.pdf