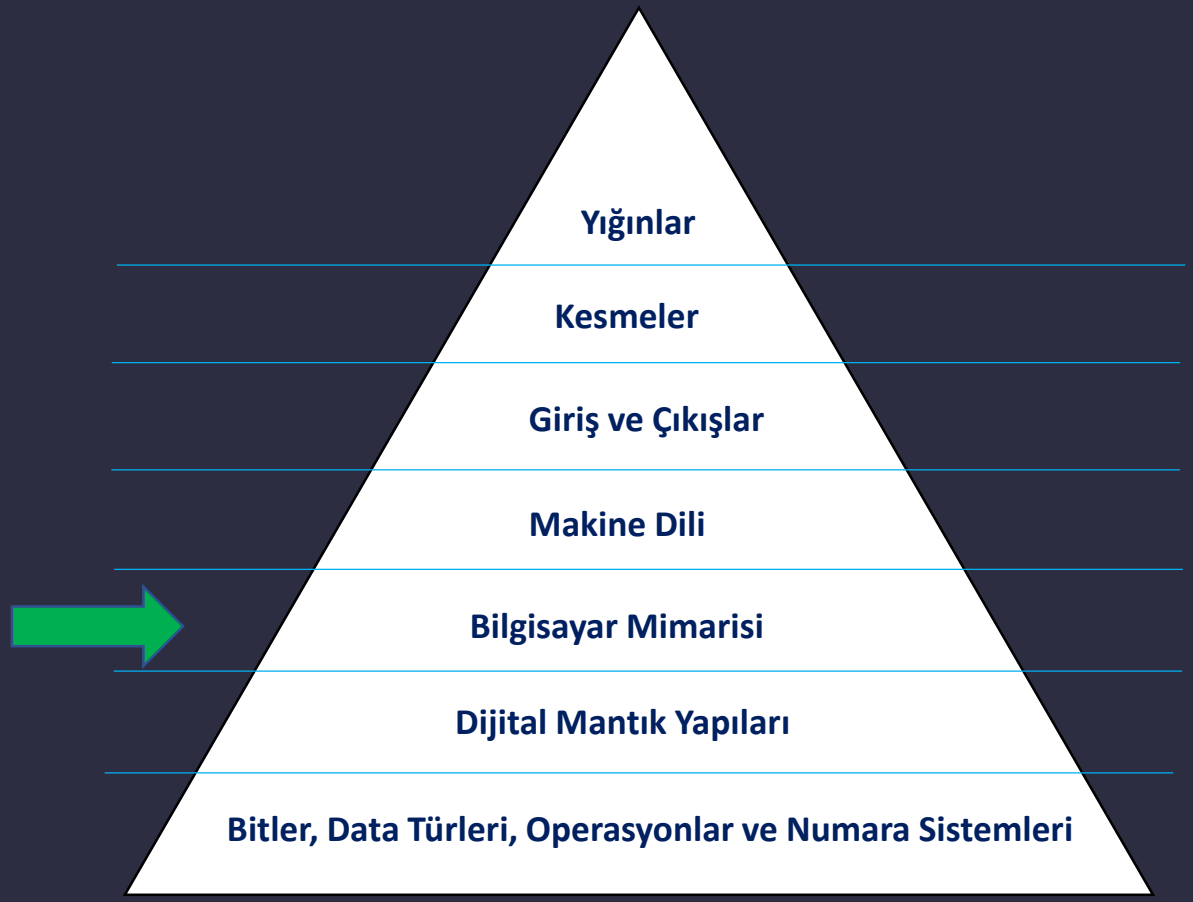
Bilgisayar Mühendisliğine Giriş – BLM 101

Hafta 8: Bilgisayar Mimarisi: Von Neuman Modeli

Fenerbahçe Üniversitesi

8. Hafta İçeriği

- Basit Bileşenler
  - Bellek
  - İşlemci Ünitesi
  - Giriş ve Çıkış
  - Kontrol Ünitesi
- LC-3 Örnek Von Neumann Makinası
- Komut İşleme
  - Komutlar
  - Komut Döngüsü
- Bilgisayar'ı Durdurma

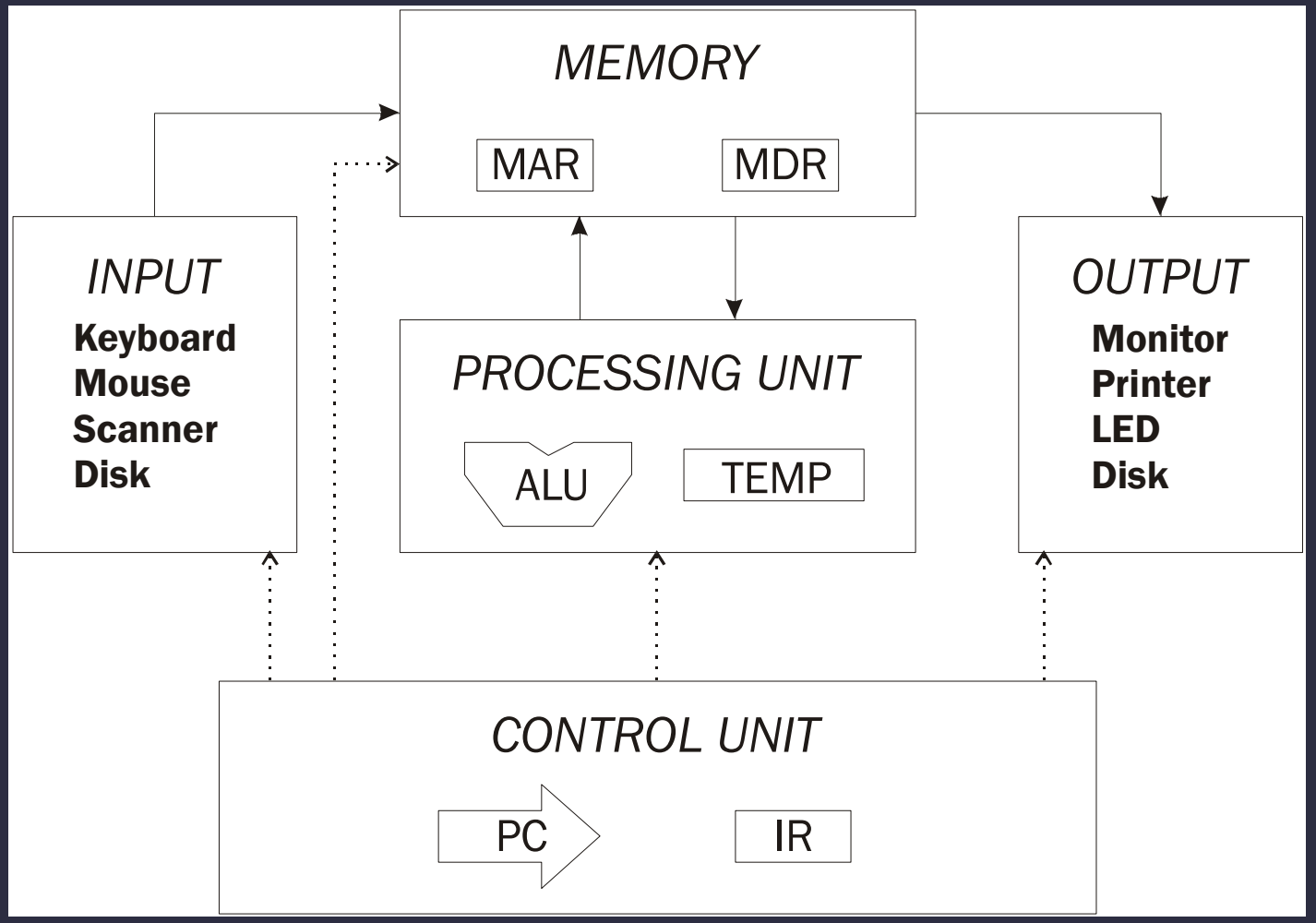


Kayıtlı Programları Yürüten Bilgisayarlar

- 1943: ENIAC
  - İlk elektronik bilgisayardır.
  - Sabit programları yürütmektedir – girişler anahtar (switchler) ile yapılmaktadır..
- 1944: EDVAC
  - Bazı geliştirmeler ile, programı bellekten yürütmektedir.
- 1945: John von Neumann
  - Kayıtlı program konsepti hakkında bir rapor yayınlamıştır.

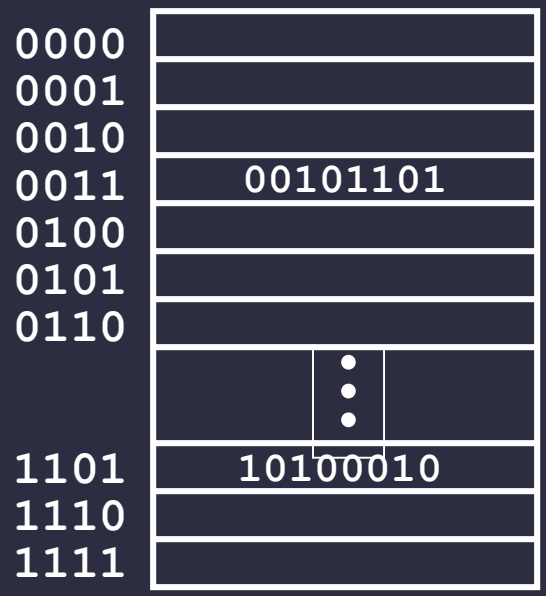
- Von Neuman makinası isminde duyurduğu mimaride, kullanılan üniteler:
  - Bellek, operasyon komutlarını ve değişkenleri tutmaktadır.
  - İşlemci Ünitesi, aritmetik ve mantık işlemlerini yapmaktadır.
  - Kontrol Ünitesi, komutların çözülmesi için gereklidir.

Von Neumann Modeli



Bellek

- 2^k x m saklama alanı vardır
- Adres
  - 2^k adet farklı saklama alanı vardır.
- İçerik
  - *Her bir saklama alanı, m bitliktir.*



- Basit operasyonlar:
- Yükleme (Load)
  - Bellekteki bir adresten verinin okunup, bir saklayıcıya yazılmasıdır.
- Kaydetme (Store)
  - Bir saklayıcıdaki içeriğin, bellekteki bir adrese yazılmasıdır.

Belleğe Erişim

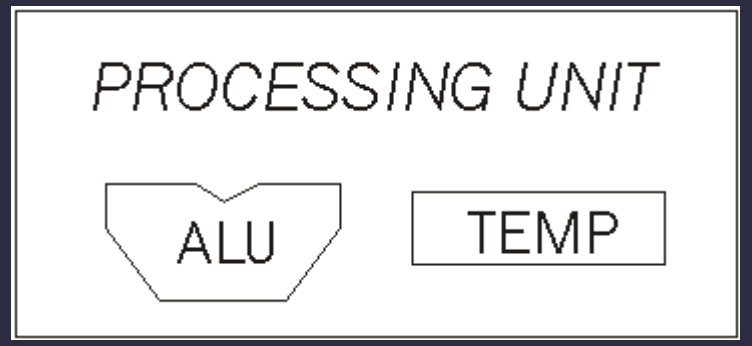
- İşlemci ünitesinin belleğe erişimi iki saklayıcı ile olmaktadır:
- MAR: Memory Address Register
- MDR: Memory Data Register

- A lokasyonundan yükleme yapmak için:
  1. MAR saklayıcısına adresi'i yazınız (A).
  2. Read sinyalini aktif edin.
  3. MDR'den gelen veriyi okuyun.

- X değerini A lokasyonuna yazmak için:
  1. X değerini MDR'e yazınız.
  2. A değerini MAR'a yazınız.
  3. Write sinyalini aktif edin.

İşlemci Ünitesi

- Fonksiyon Ünitesi
  - ALU = Aritmetik ve Lojik Ünitesi
  - Bir çok ünite bulunabilir
    (Toplama, çarpma, NOT, AND, …)
  - LC-3 isimli işlemci ADD, AND, NOT yapmaktadır.
- Saklayıcılar
  - Geçici değişkenler saklanmaktadır.
  - İşlem sonuçları tutulabilir.
  - LC-3 isimli işlemcide 8 saklayıcı ve her biri 16 bit uzunluğundadır.
- Kelime Boyutu (Word Size)
  - Saklayıcıların ve ALU'nun işlem yaptığı bit genişliğidir.

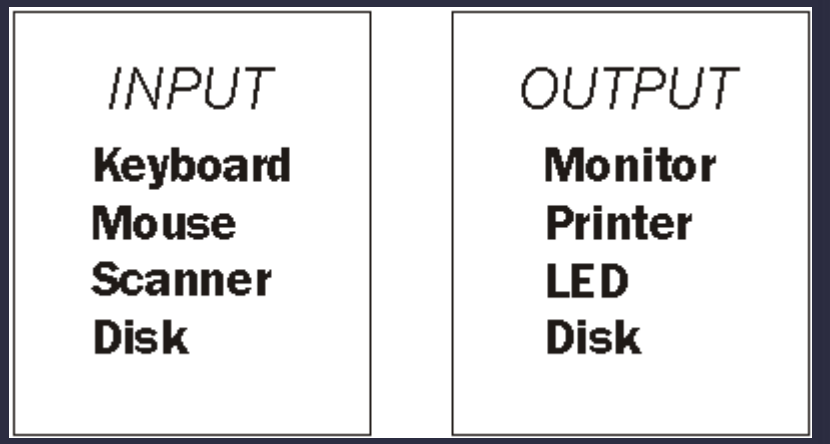


Giriş ve Çıkışlar

- İşlemciler belleğin haricinde, dışarından giriş alır ve çıkış çıkartırlar.

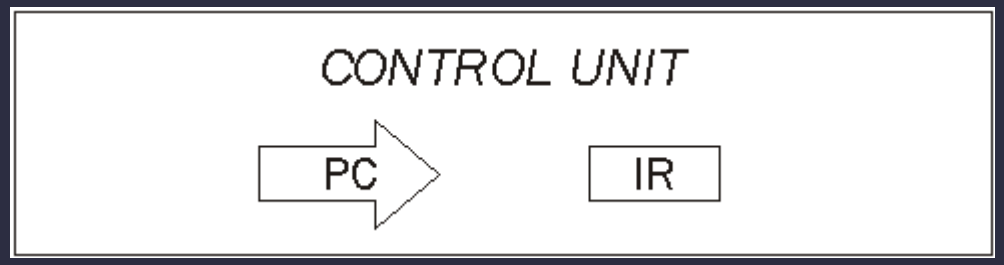
- Her bir cihaz farklı türde bir protokol ile çıktı bekler.
  - LC-3 klavye girişi ve monitör çıkışını desteklemektedir.
  - Klavye: Veri saklayıcısı (KBDR) and durum saklayıcısı (KBSR)
  - Monitör: Veri saklayıcısı (DDR) and durum saklayıcısı (DSR)

- Bazı cihazlar hem giriş alır hem de çıkış çıkartabilir.
  - Örn, sabit disk
- Programlar bu cihazları çoğunlukla "driver" denen yazılımlar ile kontrol ederler.



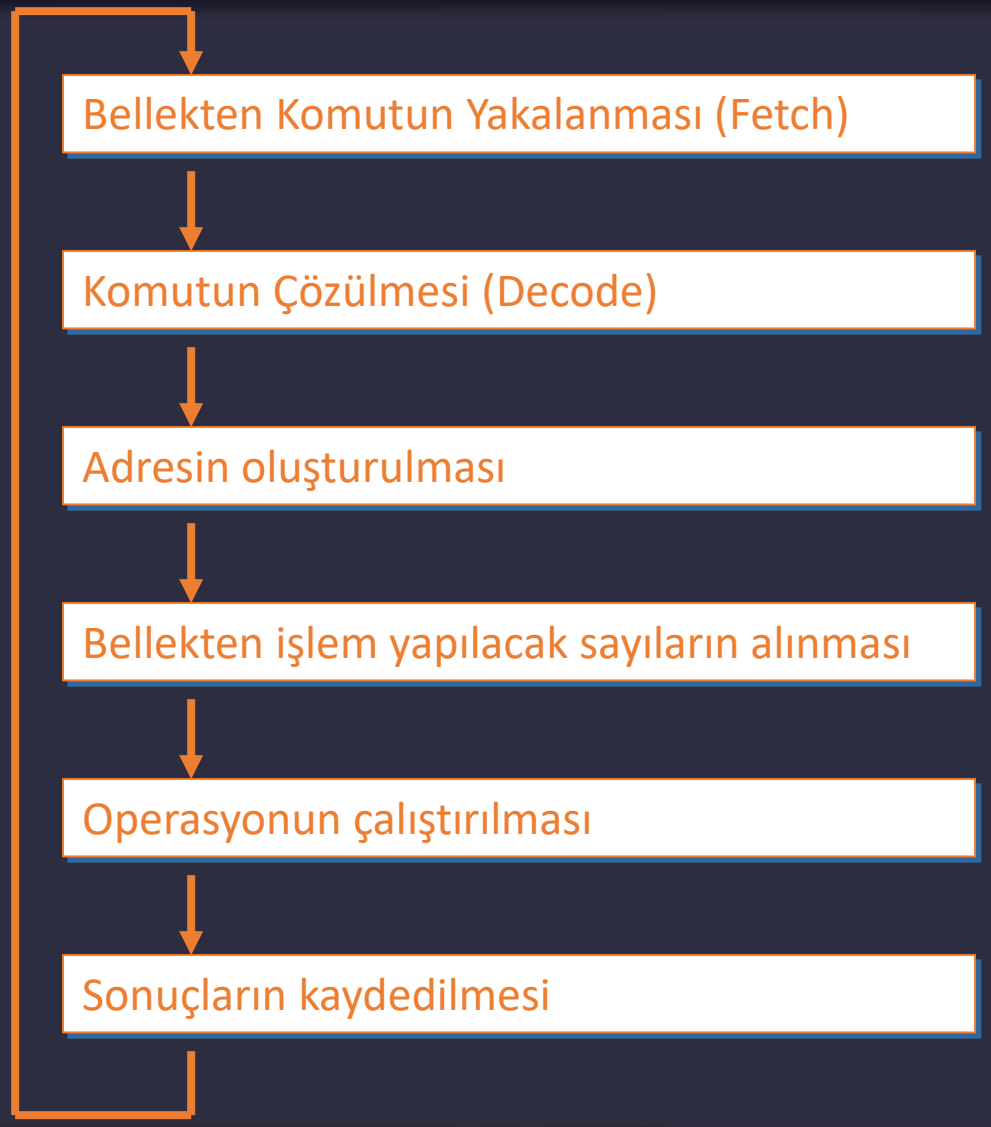
Kontrol Ünitesi

- Programın akışını yönetmektedir.



- Instruction Register (IR) şu anki koşturulan komutun adresini tutmaktadır.
- Program Counter (PC) bir sonraki koşturulacak olan komutun adresini tutar.
- Kontrol Ünitesi:
  - Bellekten program counter'ın göstediği komutu okur.
  - Sistemin geri kalanına, yapılması gereken işlemleri yaptırır.
    - Bir komut birden çok cycle sürebilir.

Komutun Çalıştırılması



Komutlar (Instruction)

- İki bilgi içermektedir:
  - *opcode*: Hangi operasyonu yapılacağını belirtir
  - *operands*: Hangi veri veya lokasyon ile işlem yapılacağını belirtir

- Bir komut ikilik tabanda 1 ve 0'lar halinde ifade edilmektedir.
  (Aynı veriler gibi)

- Bir işlemcinin çalıştırabildiği işlem komutlarının tümüne Instruction Set Architecture (ISA) denmektedir.

Örnek: LC-3 Toplama Komutu

- LC-3 komutları 16 bittir.
  - İlk 4 bit ([15:12]) operasyon kodu (opcode) dir.
- LC-3 8 adet saklayıcısı(R0-R7) vardır. Bu saklayıcılarda geçici değerleri tutmak mümkündür.
  - Toplama işleminin sayı kaynakları ve çıktısı birer saklayıcıdır.

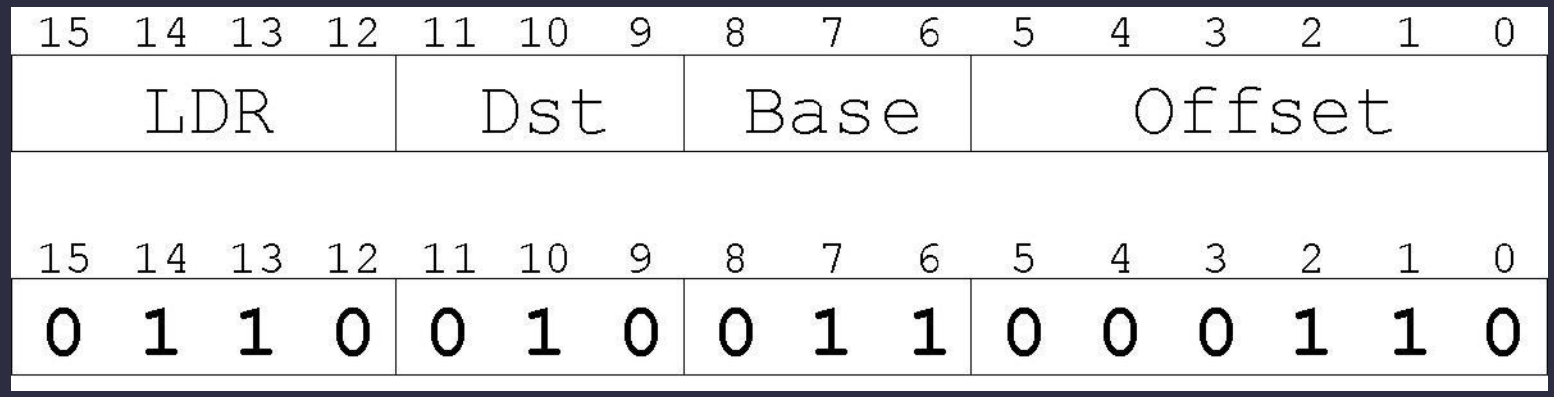
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD				Dst			Src1			0	0	0	Src2		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	1	0	0	1	0	0	0	0	1	1	0

Örn, "R2 ve R6 saklayıcılarındaki değerleri topla ve R6'a yaz"

Örnek: LC-3 LDR Komutu

- Yükleme (Load) komutu – Bellekten veri okunması
- Base + offset modu:
  - Base saklayıcısına, offset değerinin yüklenmesini sağlar
  - Bellekten veriyi alarak, hedef (destination) saklayıcısına kopyalar

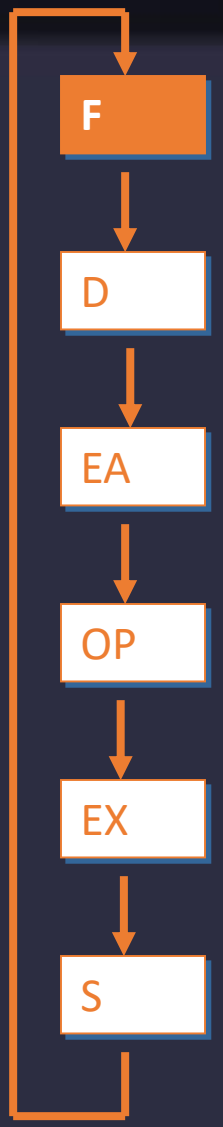


“R3 saklayıcısının içinde bulunan değere 6 eklenerek (offset), bellekten veriyi getir. Gelen veriyi R2 saklayıcısına kopyala."

Komut Yürütme: Yakalama (Fetch)

- Bir sonraki komutu bellekten alır (Program Counter'da tutulan değerdir) ve Instruction Register (IR)'a kopyalar.
  - PC'deki değeri MAR'a kopyalar.
  - Bellekteki oku sinyalini aktif et.
  - MDR'deki veriyi IR'a kopyala.

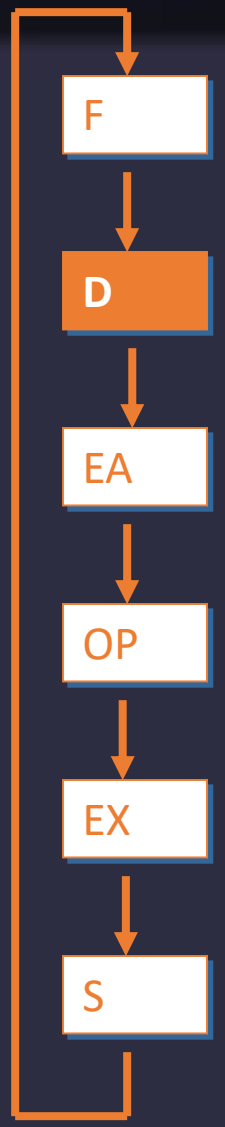
- Sonrasında PC'i bir attır.
  - PC = PC+1.



Komut Yürütme: Çözme (Decode)

- Opcode'ı çöz.
  - LC-3 işlemcisinde opcode ilk dört bittir.
  - LC-3 için 4-16 decoder devresi bu işlem için kullanılabilir.

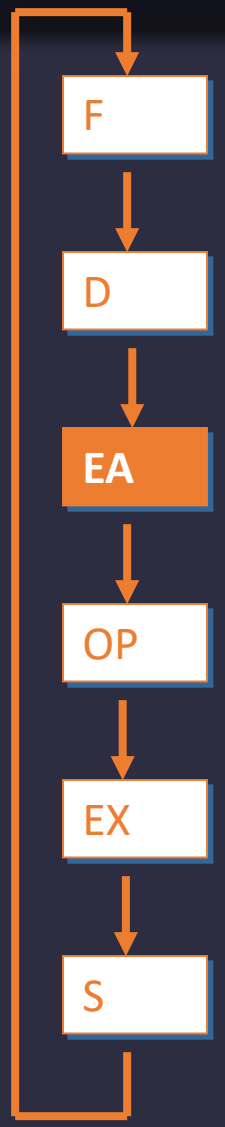
- Opcode'a göre gerekli adresleri tespit et.
  - Örnek:
    - Toplama komutu, ADD, toplanacak iki sayı ve sonucun adresini gerektirir.



Komut Yürütme: Çözme (Adresin Oluşturulması)

- Komutta erişilmesi istenen adresin hesaplandığı bölümdür.

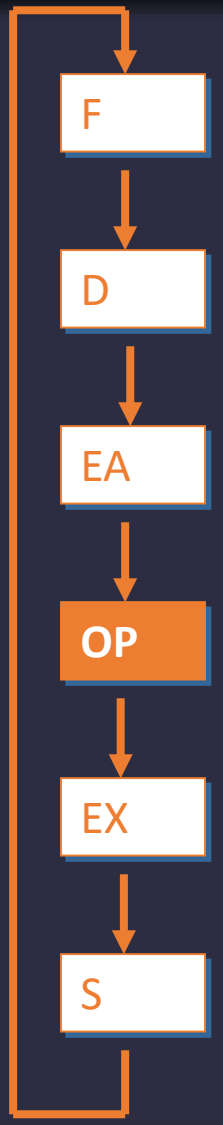
- Örnek:
  - Base saklayıcısına offset'in eklenmesi (LC-3 LDR komutu)



Komut Yürütme: Adresten Okuma Operandları Hazırlama

- İşlem yapılacak sayıların bellekten getirildiği adımdır.

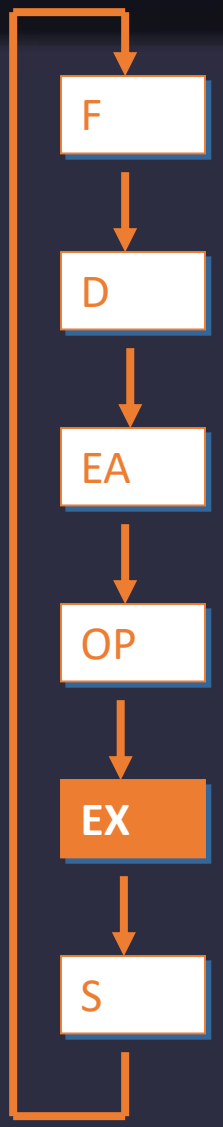
- Örnek:
  - Bellekten veri okunması (LDR)



Komut Yürütme: Çalıştırma (Execution)

- İşlenen sayılar ile operasyonu gerçekleştirir.

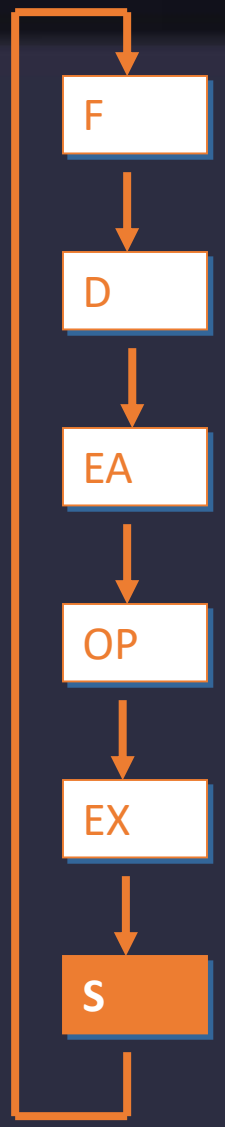
- Örnek:
  - ALU'a iki sayıyı gönderip, toplama sinyalini aktif etmek.



Komut Yürütme: Sonucu Yazma (Store)

- Hedef lokasyonuna sonucun yazılmasıdır. (Saklayıcı veya bellek olabilir)

- Örnek:
  - Toplama işleminin sonucunun bellekteki istenen yerine yazılması.



Komutların Akış Sırasını Değiştirmek

- Yakalama (Fetch) aşamasında PC'İ (Program Counter) bir arttırılmaktadır.
- Programın içindeki bir başka yere atlamak istendiğinde ne yapılması gerekmektedir.
  - Örnek: Döngüler, fonksiyon çağrılması
- PC'nin değerini değiştirecek özel komutlara ihtiyaç duyulmaktadır.
- Bu komutlara kontrol komutları denir.
  - Zıplama (jump) koşulsuz olarak PC'i istenen değer ile değiştirir.
  - Dallanmalar (branches) koşula bağlı olarak çalışırlar – istenen koşul doğru ise PC'nin değerini değiştirirler.

Örnek: LC-3 JMP Komutu

- Bir saklayıcıdan PC'nin değerini güncelle. Bir sonraki işletilecek olan kod artık saklayıcının içindeki değerden alınacaktır.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JMP				0	0	0	Base			0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0

"R3 saklayıcısının içindeki değeri PC'e ata"

Komut İşleme

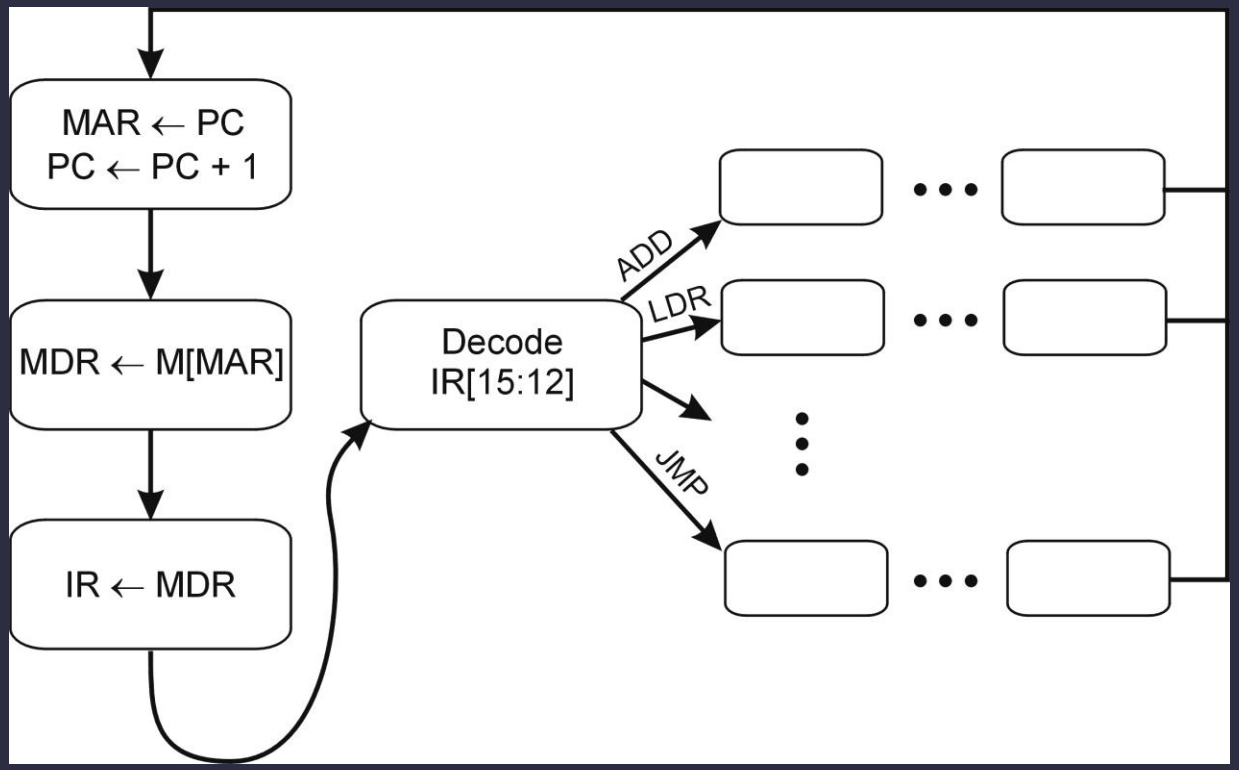
- Tüm komutlar 1-0'lardan oluşurlar, aynı veriler gibi

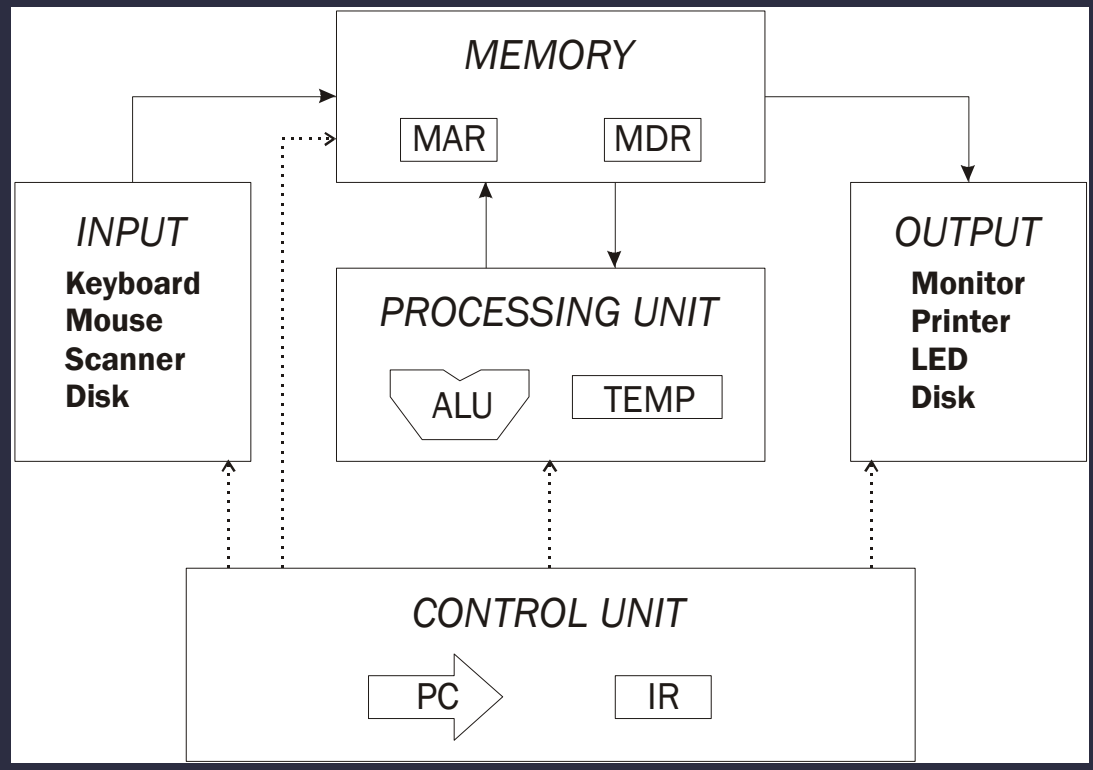
- Üç temel komut bulunmaktadır:
  - Hesaplama komutları (ADD, AND, …)
  - Veri hareketi komutları(LOAD, STORE, …)
  - Kontrol komutları (JMP, BRnz, …)

- Bir komutun çalıştırılması için gereken 6 temel adım vardır:
- $F \rightarrow D \rightarrow EA \rightarrow OP \rightarrow EX \rightarrow S$
  - Bazı adımlar bir cycle sürerken, bazı adımlar birden fazla cycle sürebilir.

Kontrol Ünitesinin Durum Diyagramı

- LC-3 işlemcisi için örnek durum diyagramı:





Saat (Clock)'u Durdurmak

- Kontrol ünitesi, kendine gelen clock sinyali oldukça işlem yapmaya devam edecektir.

- İşlemciyi durdurmak:
  - Clock üretici (Kristal)'nin çıkarttığı sinyali 0 ile AND işlemi yapın.
  - Böylelikle kontrol ünitesine clock sinyali sürekli 0 gidecek, hiç yükselen kenar almayacağı için işlem yapmayacaktır.