Bölüm 1: Giriş ve Temel Kavramlar

Başlık: Makine Dili ve Bilgisayar Mimarisi

Alt Başlık: Brookshear Sanal Makinesi ve Veri Manipülasyonu

Ders: Bilgisayar Bilimlerine Giriş

Hazırlayan: Barış Çelikel 25360859093

BURSA TEKNİK ÜNİVERSİTESİ

Sunum İçeriği

- **Bilgisayar Mimarisi (Sec 2.1): CPU, ALU, Kontrol Birimi, Registerlar.**

- **Makine Dili (Sec 2.2): Komut Seti Mimarisi ve Komut Formatı.**

- **Brookshear Makinesi (Appendix C): 12 Temel Komutun Detaylı İncelemesi.**

- **Program Yürütme (Sec 2.3): Makine Döngüsü (Fetch-Decode-Execute).**

Bilgisayarın Temel İşlevi: Veri Manipülasyonu

- **Bilgisayarlar temelde verileri alır, işler (manipüle eder) ve çıktı üretir.**

- **Bu işlemler donanım seviyesinde elektrik sinyalleri (0 ve 1) ile gerçekleşir.**

- **Temel Soru: Donanım, yazılımın isteklerini nasıl anlar ve uygular?**

  - ✓ **Cevap: Makine Dili ve Mimarisi.**

BÖLÜM 2: BİLGİSAYAR MİMARİSİ

Bilgisayarın Kalbi: CPU (Merkezi İşlem Birimi)

- **Tanım: Bilgisayarın işlemlerini yürüten ve yöneten ana parçadır ("Beyin").**

- **İki ana bölümden oluşur:**
  1. **Aritmetik/Mantık Birimi (ALU)**
  2. **Kontrol Birimi (Control Unit)**

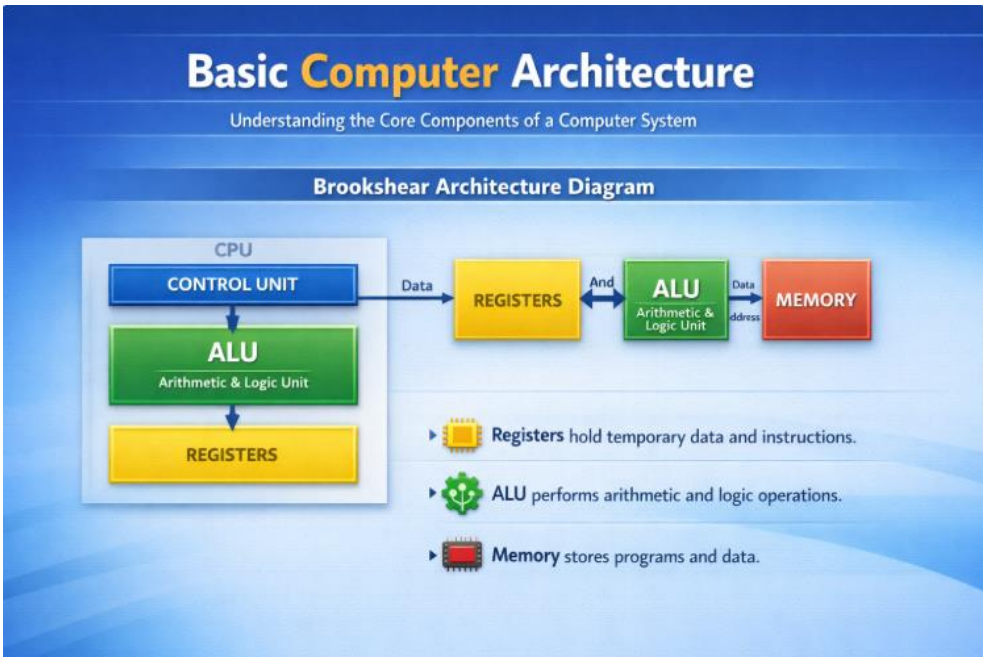
- **CPU, Ana Bellek (RAM) ile sürekli iletişim halindedir.**

Aritmetik Mantık Birimi (ALU)

- **Görevi: Veriler üzerinde hesaplama ve mantıksal karşılaştırma işlemlerini yapar.**
- **İşlem Türleri:**

    1. **Aritmetik: Toplama, Çıkarma (Çarpma ve bölme genellikle tekrarlı toplama ile yapılır).**
    2. **Mantıksal: AND (VE), OR (VEYA), XOR, NOT, Shift (Kaydırma).**

- **ALU, veriyi Register'lardan alır, işler ve sonucu tekrar Register'a yazar.**



Kontrol Birimi (Control Unit)

- **Görevi: Bilgisayarın "Trafik Polisi"dir.**

- **İşlemleri kendisi yapmaz, diğer birimlerin ne zaman ne yapacağını yönetir.**

- **Fonksiyonları:**
  - **Bellekten komutları getirir.**
  - **Komutları çözer (Decode).**
  - **ALU ve Bellek birimlerine gerekli sinyalleri gönderir.**

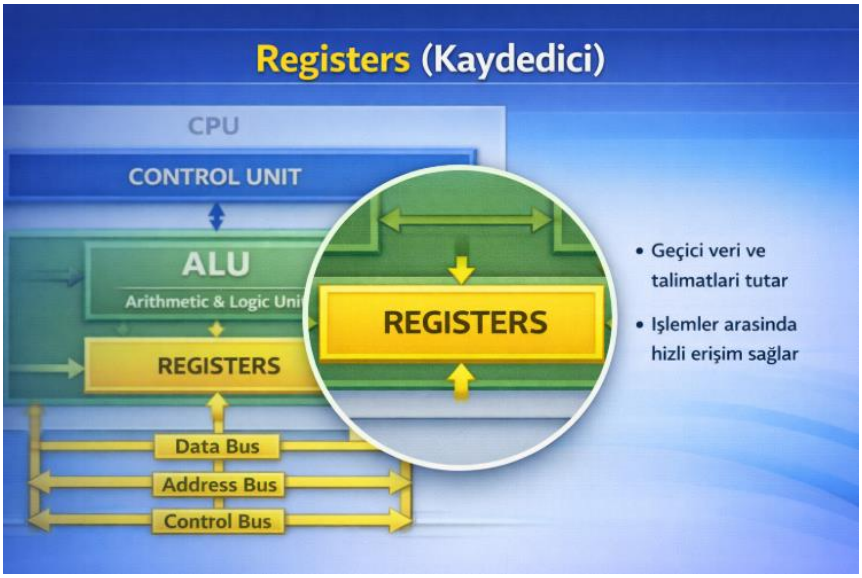
Kaydediciler (Registers)

- **Nedir? CPU'nun içinde bulunan, çok hızlı ve geçici veri saklama alanlarıdır.**

- **Neden Gerekli? Ana bellek (RAM) CPU'nun hızına yetişemez. İşlem yapılacak veriler önce Register'lara alınır.**

- **Brookshear Makinesinde:**

  - **Genel Amaçlı Kaydediciler: 16 Adet (R0, R1, ... RF). Veri manipülasyonu için kullanılır.**
  - **Özel Amaçlı Kaydediciler: PC ve IR (Bir sonraki slaytta detaylandırılacak).**



Özel Amaçlı Kaydediciler

- **Bu kaydediciler programcı tarafından veri saklamak için değil, işlemcinin takibi için kullanılır.**

- **1. Program Sayacı (Program Counter - PC): Bir sonraki çalıştırılacak komutun bellek adresini tutar.**

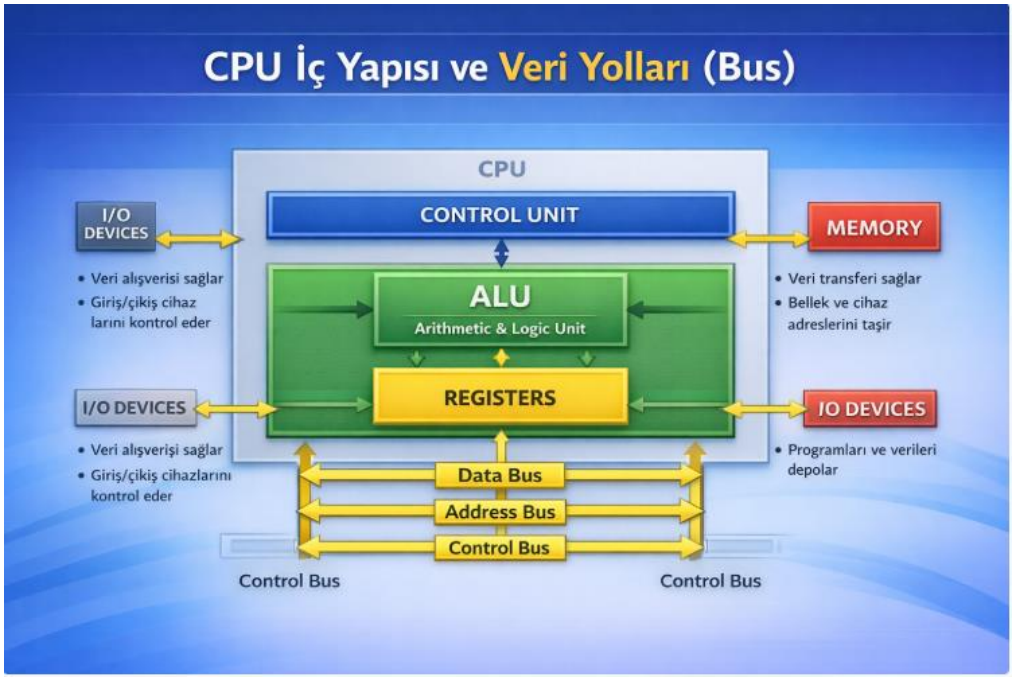
- **2. Komut Kaydedicisi (Instruction Register - IR): Bellekten o an getirilmiş olan ve üzerinde çalışılan komutu tutar.**

Veri Yolu (Bus) ve Ana Bellek İlişkisi

- **CPU ve Bellek arasındaki iletişim BUS adı verilen kablo hatlarıyla sağlanır.**

- **Adres Yolu, Veri Yolu ve Kontrol Yolu olarak ayrılır.**

- **CPU, bellekteki bir veriyi okumak için o verinin adresini Bus'a koyar, bellek de veriyi Bus üzerinden CPU'ya gönderir.**



BÖLÜM 3: MAKİNE DİLİ

Makine Dili (Machine Language)

- **Bilgisayar donanımının doğrudan anlayabildiği tek dildir.**
- **Yapısı: Tamamen 0 ve 1'lerden (Bit) oluşur.**
- **Instruction Set (Komut Seti): Bir işlemcinin tanıdığı ve yürütebildiği tüm komutların listesidir.**
- **Önemli: Makine dili donanıma bağımlıdır (Intel x86 makine dili ile ARM veya Brookshear makine dili farklıdır).**

Brookshear Komut Formatı

- **Brookshear mimarisinde her komut 16 bit (2 Byte) uzunluğundadır.**

- **İnsanların okumasını kolaylaştırmak için genellikle Hexadecimal (Onaltılık) formatta gösterilir.**

- **Örnek Komut: 156C (Hex) = 0001 0101 0110 1100 (Binary)**

Komutun Anatomisi: Op-code ve Operand

- **Bir makine komutu iki ana kısımdan oluşur:**

- **1. Op-code (Operation Code):**
  - **"Ne yapılacak?" sorusunun cevabıdır.**
  - **Brookshear makinesinde komutun ilk 4 biti (ilk Hex hanesi) Op-code'dur.**
  - **Örn: 156C -> 1 (s işlemi).**

- **2. Operand (İşlenen):**
  - **"Kiminle yapılacak?" sorusunun cevabıdır.**
  - **Geriye kalan 12 bit (son 3 Hex hanesi) Operand'dır.**
  - **Adresleri, register numaralarını veya sabit sayıları belirtir.**

Depolanmış Program Kavramı (Stored-Program Concept)

- **Geçmişte programlar kablolarla donanımsal olarak ayarlanıyordu.**

- **Günümüzde Programlar (kodlar) ve Veriler aynı bellekte saklanır.**

- **Bu sayede, sadece bellekteki yazılımı değiştirerek bilgisayarın görevi değiştirilebilir.**

- **Bu kavram modern bilgisayarların esnekliğinin temelidir.**

BÖLÜM 4: BROOKSHEAR MAKİNESİ KOMUTLARI

Veri Transferi - LOAD (Op-code 1)

- **Komut: 1RXY**
- **Anlamı: Bellek hücresi XY adresindeki içeriği al, R numaralı register'a yükle.**
- **Detay: Bellekteki veri silinmez, kopyalanır.**
- **Örnek: 14A3 -> Bellek A3 adresindeki veriyi Register 4'e kopyala.**

Veri Transferi - LOAD IMMEDIATE (Op-code 2)

- **Komut: 2RXY**
- **Anlamı: XY sabit değerini (deseni) R numaralı register'a yükle.**
- **Önemli Fark: Op-code 1 belleğe gider, Op-code 2 ise komutun içindeki sayıyı doğrudan alır.**
- **Örnek: 2B05 -> Register B'nin içine "05" sayısını yükle.**

Veri Transferi - STORE (Op-code 3)

- **Komut: 3RXY**
- **Anlamı: R numaralı register'ın içeriğini bellek hücresi XY adresine kaydet.**
- **Detay: Bellek hücresindeki eski veri silinir, yerine yenisi yazılır (Yıkıcı işlem).**
- **Örnek: 3510 -> Register 5'teki veriyi Bellek 10 adresine yaz.**

Veri Transferi - MOVE (Op-code 4)

- **Komut: 40RS**
- **Anlamı: R numaralı register'ın içeriğini S numaralı register'a kopyala.**
- **Not: 0 kısmı kullanılmaz, dolgu malzemesidir.**
- **Örnek: 40A2 -> Register A'daki veriyi Register 2'ye kopyala.**

Aritmetik İşlemler - ADD Integer (Op-code 5)

- **Komut: 5RST**
- **Anlamı: Register S ve Register T içindeki sayıları Tamsayı (İkinin Tümleyeni) olarak topla, sonucu Register R'ye yaz.**
- **Örnek: 5723 -> R2 + R3 işlemini yap, sonucu R7'ye yaz.**
- **Not: Toplama işlemi ALU tarafından gerçekleştirilir.**

Aritmetik İşlemler - ADD Float (Op-code 6)

- **Komut: 6RST**

- **Anlamı: Register S ve Register T içindeki sayıları Kayan Noktalı (Floating Point) olarak topla, sonucu Register R'ye yaz.**

- **Önem: Bilgisayarlar tamsayıları ve ondalıklı sayıları farklı formatlarda saklar ve işler. Bu komut ondalıklı toplama içindir.**

Mantıksal İşlemler - OR (Op-code 7)

- **Komut: 7RST**
- **Anlamı: S ve T registerları üzerinde mantıksal OR işlemi yap, sonucu R'ye yaz.**
- **Kullanım: Genellikle belirli bitleri "1" yapmak (Setting bits) için kullanılır.**
- ***Örnek:* 1010 OR 0011 -> 1011**

Mantıksal İşlemler - AND (Op-code 8)

- **Komut: 8RST**
- **Anlamı: S ve T registerları üzerinde mantıksal AND işlemi yap, sonucu R'ye yaz.**
- **Kullanım: "Maskeleme" (Masking) için kullanılır. Bir verinin sadece belirli bir kısmını alıp diğerlerini sıfırlamak için idealdir.**
- ***Örnek:* 1011 AND 0001 -> 0001**

Mantıksal İşlemler - XOR (Op-code 9)

- **Komut: 9RST**
- **Anlamı: S ve T registerları üzerinde mantıksal XOR (Özel Veya) işlemi yap.**
- **Özellik: Eğer iki bit aynıysa 0, farklıysa 1 üretir.**
- **Kullanım: Bir register'ı kendisiyle XOR'larsanız sonuç daima 0 olur (Register sıfırlama).**

Manipülasyon - ROTATE (Op-code A)

- **Komut: AR0X**
- **Anlamı: Register R'in içeriğini X kez sağa döndür (Rotate Right).**
- **Çalışma Prensibi: En sağdan çıkan bit kaybolmaz, en başa (en sola) geçer.**
- **Örnek: A301 -> R3'ü 1 bit sağa döndür.**

Kontrol Komutları - JUMP & HALT (Op-code B & C)

- **JUMP (B): BRXY**
    - **Koşullu Dallanma: Eğer Register R ile Register 0'ın içeriği EŞİTSE, program XY adresine atlar. Değilse sıradaki komuttan devam eder.**
    - **Döngü (Loop) ve If-Else yapıları bu komutla kurulur.**
- **HALT (C): C000**
    - **Programı durdurur. İşlemci çalışmayı bırakır.**

BÖLÜM 5: PROGRAMIN YÜRÜTÜLMESİ

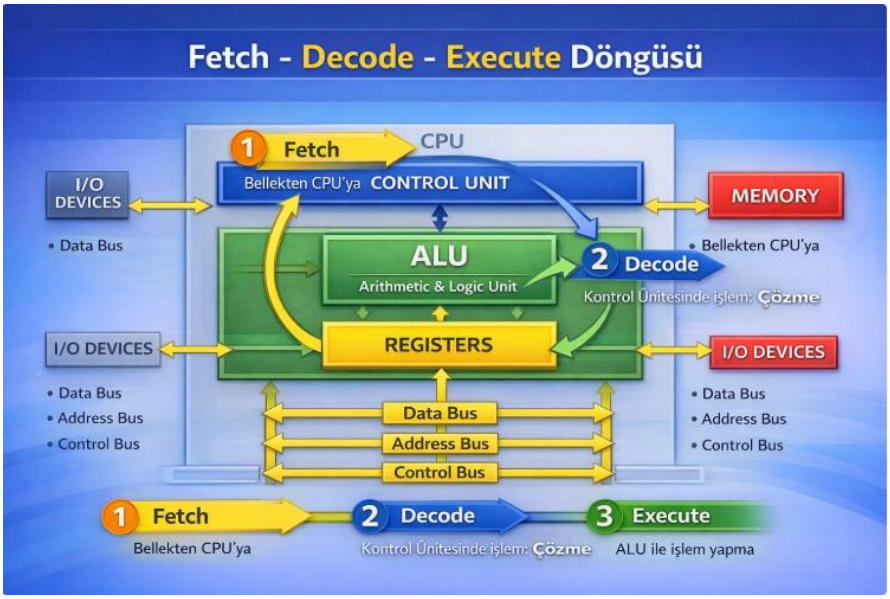
Makine Döngüsü (The Machine Cycle)

Bilgisayar bir saat (Clock) tarafından yönetilen ritmik bir döngü içinde çalışır. Bu döngü üç ana adımdan oluşur:

1. **Fetch (Getir)**

2. **Decode (Çöz)**

3. **Execute (Yürüt)**



Adım 1 - Getir (Fetch)

- **Kontrol birimi, Program Sayacı (PC) içindeki adrese bakar.**
- **O adresteki komutu bellekten okur.**
- **Okunan komutu Komut Kaydedicisine (IR) yerleştirir.**
- **Çok Önemli: PC değerini bir artırarak bir sonraki komuta hazırlar.**

Adım 2 - Çöz (Decode)

- **Komut Kaydedicisindeki (IR) 16 bitlik veri analiz edilir.**
- **İlk 4 bite (Op-code) bakılarak ne yapılacağı anlaşılır.**
- **Örneğin Op-code 5 ise ALU'ya "toplama yapmaya hazırlan" sinyali gider.**
- **Op-code 1 ise bellek birimine "veri okumaya hazırlan" sinyali gider.**

Adım 3 - Yürüt (Execute)

- **Kontrol birimi işlemi tetikler.**
- **Veri register'dan ALU'ya gider, toplanır ve geri döner; veya bellekten veriler transfer edilir.**
- **İşlem bittiğinde döngü başa döner (Fetch).**

Örnek Senaryo İzleme

Program: İki sayıyı toplayan kod.

1. **156C: (Fetch) 6C adresindeki sayıyı al -> (Decode) Yükle -> (Execute) R5'e koy.**

2. **166D: (Fetch) 6D adresindeki sayıyı al -> (Decode) Yükle -> (Execute) R6'ya koy.**

3. **5056: (Fetch) R5 ve R6'yı al -> (Decode) Topla -> (Execute) Sonucu R0'a yaz.**

4. **C000: (Fetch) -> (Decode) Dur -> (Execute) Sistem durur.**

BÖLÜM 6: KAPANIŞ

Özet ve Sonuç

- **Bilgisayarlar, karmaşık işlemleri çok basit temel komutları (Yükle, Topla, Kaydet) çok hızlı arka arkaya yaparak gerçekleştirir.**

- **Makine dili, donanım ile yazılım arasındaki köprüdür.**

- **Brookshear mimarisi, modern bilgisayarların çalışma prensibini anlamak için basitleştirilmiş mükemmel bir modeldir.**

- **Dinlediğiniz için teşekkürler.**