BİLGİSAYAR MİMARİSİ VE ORGANİZASYONU

1. BÖLÜM

Bilgisayarın Fonksiyon ve Arabağlantılarına Genel Bakış (Sistem Ortak Yolu)

ÖZET:

1.Bir komut çevrimi, bir komutun alınması, bunu izleyen 0 veya daha fazla sayıda operand'ın alınması, bunu izleyen 0 veya daha fazla sayıda operand'ın depolanması ve bunu izleyen (eğer kesmeler aktif ise) bir kesme kontrolünden oluşur.

2.Bilgisayar sisteminin temel elemanlarının (işlemci, ana bellek ve Giriş/Çıkış) veri ve kontrol sinyallerini birbirleri arasında iletebilmeleri için bağlantılar olmalıdır. Bu bağlantıları oluşturmada en sık kullanılan yöntem, birden fazla hattı olan paylaşımlı ortak yol sistemidir (bus system). Günümüzdeki sistemlerde, performansı artırmak amacıyla, ortak yollar arasında hiyerarşik bir düzen bulunur.

3.Bir ortak yol sistemi ile ilgili olarak üzerinde durulması gereken konular, karar verme düzeneği, zamanlama ve ortak yol genişliğidir. (1) Karar verme düzeneği, ortak yol hatlarından sinyal göndermek için gerekli iznin merkezden mi verileceği yoksa diğer birimlerden mi verileceği ile ilgilidir. (2) Zamanlama, ortak yol üzeindeki sinyallerin merkezi bir saat ile mi yoksa en son gerçekleştirilen iletime bağlı olarak asenktron mu yapılacağı ile ilgilidir. (3) ortak yol genişliği adres ve veri ortak yollarının hat (bit) sayısı ile ilgilidir.

Genel olarak bakıldığında, herbirinden bir ya da daha fazlası olmak üzere CPU, bellek ve Giriş/Çıkış elemanları bir bilgisayarı oluşturan bileşenlerdir. Bunların hepsi bir bilgisayarın temel fonksiyonu olan programları çalıştırma görevini yerine getirmek için birbirleriyle bir şekilde bağlantılıdırlar. O halde, yine genel bir bakışla bir bilgisayar sistemi aşağıda verilen yollarla tarif edilebilir: (1) Her bileşenin diğer bileşenlere gönderdiği ya da onlardan gelen veri ve kontrol sinyallerini tanımlayarak, ve (2) Ara bağlantı yapısını ve ara bağlantı yapısını yöneten kontrol sinyallerini tanımlayarak.

Yapı ve fonksiyonların bu şekilde genel olaral anlaşılması önemlidir. Çünkü bunlar bir bilgisayarın doğasının anlaşılmasını sağlarlar. Böylelikle sistemin işlemesini yavaşlatan durumların, alternatif çözümlerin ve eğer bir bileşen başarısız olursa bunun sistem için ne büyüklükte bir sorun olduğunun tesbitini ayrıca performansı artırmadaki kolaylıkların görülmesi sağlanır.

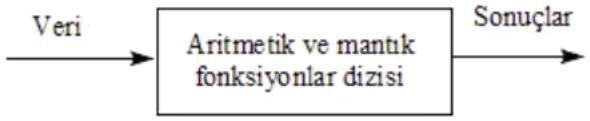
1.1 Bilgisayarın Bileşenleri

Bütün bilgisayar tasarımları John von Neumann tarafından geliştirilen kavramlara dayanır. Bu yüzden von Neumann mimarisi olarak adlandırılan yapı üç temel kavrama dayanır:

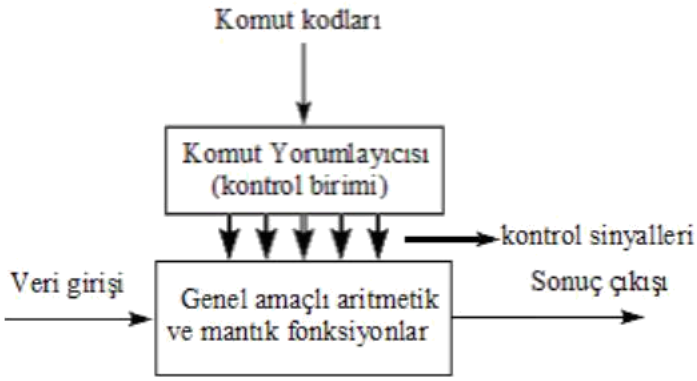
- Bütün veri ve komutlar tek bir okunabilir-yazılabilir bellekte saklanır.
- Bu belleğin içeriğindekilerin ne tür veri olduğuna bakılmaksızın sadece adres belirtmek suretiyle içeriğe erişilebilir.
- Komutlar sırayla çalıştırılırlar.

Aritmetik ve lojik işlemleri yapmakta ve veri depolamakta kullanılabilen lojik elemanlar vardır. Eğer özel bir tip hesaplama işlemi gerçekleştirilecekse bu lojik elemanlar, bu işe özel olarak birleştirilerek işin yapılması sağlanabilir. Lojik elemanların bu şekilde istenen sırayla yerleştirilmesi ve özel bir hesaplama işleminin yaptırılması bir tür "programlama" olarak düşünülebilir. Bu türdeki "program" *donanımsal program – hardwired program* olarak adlandırılır.

Alternatif olarak, aritmetik ve lojik fonksiyonları gerçekleştirebilen genel amaçlı bir yapı oluşturulabilir. Böyle bir donanım, sağlanan kontrol sinyallerine bağlı olarak veri üzerinde farklı işlemleri gerçekleştirebilir. İlk durumda, sistem veriyi kabul eder ve sonuçları üretir(Şekil 1.1a). Genel amaçlı donanım yapısında ise sistem veriyi ve kontrol sinyallerini alır, sonuçları üretir. Böylelikle her yeni program için yeni bir devre bağlantısı oluşturmak yerine, programcı sadece yeni bir dizi kontrol sinyali verilmesini sağlar.



(a) Donanım ile programlama



(b) Yazılım ile programlama

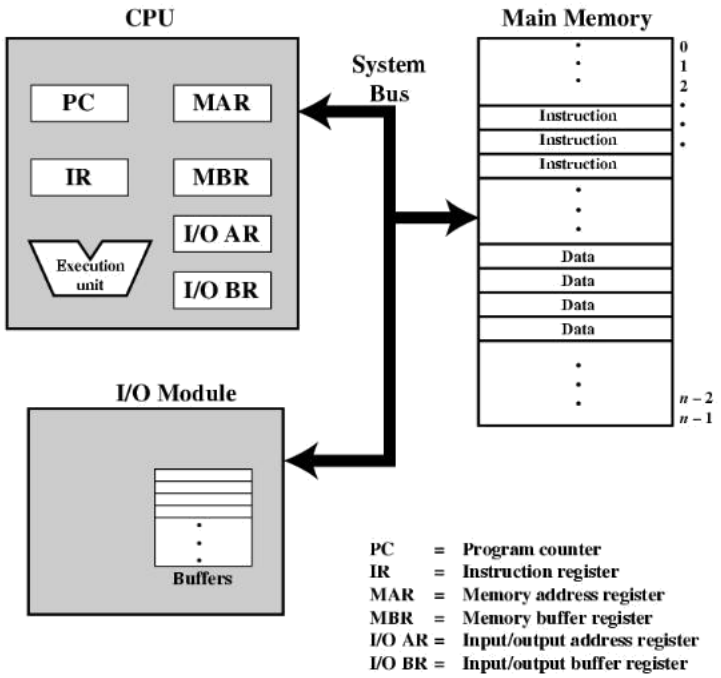
Şekil 1.1 Donanım ve Yazılım Yaklaşımları

Bu konrol sinyalleri nasıl verilecektir? Aslında programın tamamı ardı ardına sıralanmış adımlardan oluşur. Bu adımların herbiri ile veriler üzerinde aritmetik veya mantıksal işlemler yapılır. Herbir adım için yeni bir dizi kontrol sinyaline ihtiyaç duyulur. Gerekli olabilecek herbir kontrol sinyali dizisi için özel bir kod belirlenir ve bu kodları anlayarak gerekli kontrol sinyallerini üretebilecek genel amaçlı donanım yapısı sisteme eklenir(Şekil 1.1b).

Şimdi programlama çok daha kolaylaştı. Her yeni program için yeni bir donanım oluşturmak yerine artık sadece yeni bir kodlar dizisi oluşturmak yeterlidir. Her kod aslında bir komuttur ve donanım tarafından yorumlanarak gerekli kontrol sinyalleri oluşturulur. Bu kodlar ya da komutlar dizisi "yazılım" olarak adlandırılır.

Şekil 1.1b sistemin iki ana bileşenini gösteriyor: bir komut yorumlayıcısı ve aritmetik ve mantık işlemleri yerine getirecek genel amaçlı donanım yapısı. Bu iki bileşen CPU'yu (Merkezi İşlem Birimi) oluşturur. Bir bilgisayar için hala birkaç bileşene ihtiyaç vardır. Örneğin veri ve komutların sisteme alınması için bir tür giriş birimine. Bu birim, veri ve komutları bir şekilde alarak onları sistem tarafından kullanılabilecek sinyallere dönüştürecektir. İşlem sonuçlarının sistem dışına verilebilmesi için de bir tür çıkış birimine ihtiyaç vardır. Bu birimler de birlikte Giriş/Çıkış (Input/Output) olarak adlandırılır.

Ayrıca komutları verileri geçici olarak saklayacak bir birime daha ihtiyaç vardır. Bu birim, harici bellek ve diğer çevre birimlerinden ayrılabilmesi için "ana bellek" olarak adlandırılır. Von Neumann hem veri hem de komutların aynı birimde saklanabileceğini belirtmiştir.



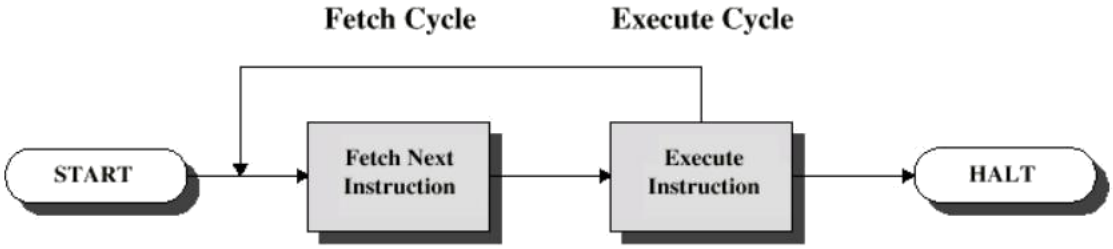
Şekil 1.2 Bilgisayarın ana bileşenleri

Şekil 1.2 bilgisayarın ana bileşenlerini ve aralarındaki bağları gösteriyor. CPU bellek ile veri alışverişinde bulunur. Bunun için iki register'ı vardır: MAR (memory address register) bellekte okunacak veya yazılacak bellek bölgesinin adresini saklar. MBR ise (memory buffer register) belleğe yazılacak olan veriyi veya bellekten okunacak olan veriyi saklar. Benzer olarak G/Ç adres register'ı (I/O AR) belirli bir giriş veya çıkış birimini işaret eder, G/Ç buffer register (I/O BR) ise CPU ile G/Ç birimi arasındaki veri alışverişi için kullanılır.

Şekil 1.2 de bellek birimi de görülüyor. Belleğin herbir satırında veri ya da komut olabilen ikili sistem (binary) sayılar bulunur. Bir G/Ç birimi ise dış cihazlardan CPU ve belleğe, ya da ters yöne, veri taşır. G/Ç birimi içinde veri yerine gönderilinceye kadar geçici olarak tutulduğu buffer'lar vardır.

1.2 Bilgisayarın Görevi

Bir bilgisayarın görevi, bellekte bir dizi komut halinde saklanmş olan, programı çalıştırmaktır. Bir komutu çalıştırılması, en basit haliyle, şu iki adımdan oluşur: komutun bellekten alınması ve sonra yerine getirilmesi. Bütün komutların sırayla bu şekilde çalıştırılması ile program çalıştırılmış olur. Tek bir komutun çalıştırılması bir "komut çevrimi--instruction cycle" olarak adlandırılır. Biraz evvel bahsedilen iki adım ise "gidip-getirme çevrimi—fetch cycle" ve "yürütme çevrimi—execution cycle" olarak adlandırılır. Programın çalışması ancak makinenin kapatılmasıyla, düzeltilemeyen bir hatanın oluşmasıyla veya programı durduracak bir komutun çalıştırılmasıyla durdurulabilir.



Şekil 1.3 Komut çevrimi

Program Kavramı

- Sabit bağlantılı sistemler esnek değildir.
- Genel amaçlı donanımlar doğru kontrol sinyalleri verildiğinde farklı görevleri yerine getirebilirler.
- Bağlantıları tekrar yapmak yerine yeni bir dizi kontrol sinyali yeterli olur.

Program Nedir?

- Bir dizi işlemdir.
- Her adımda bir aritmetik veya mantıksal işlem (yani karar verme işlemi)yapılır.
- Her bir işlem için farklı bir dizi kontrol sinyali gerekir.

Kontrol Biriminin Görevi

- Her bir işlem için belirli özel bir kod vardır
-- Örneğin ADD, MOV
- Donanımın bir parçası kodu kabul eder ve ilgili kontrol sinyallerini üretir
- Artık bir bilgisayarımız var.

Bileşenler

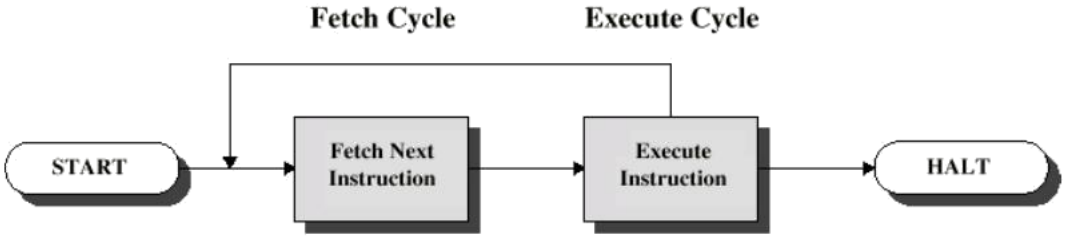
— Kontrol birimi ve aritmetik lojik birimi Merkezi İşlem Birimini (CPU) oluştururlar

— Veri ve komutların sisteme girilmesi ve sonuçlarında sistemden gönderilmesi gerekir. Yani: Giriş/Çıkış

— Kod ve sonuçların geçici olarak depolanması gerekir. Yani: Ana bellek

Komut çevrimi

- İki adımdan oluşur:
  - — Fetch -gidip getirme
  - — Execute –yerine getirme(çalıştırma)



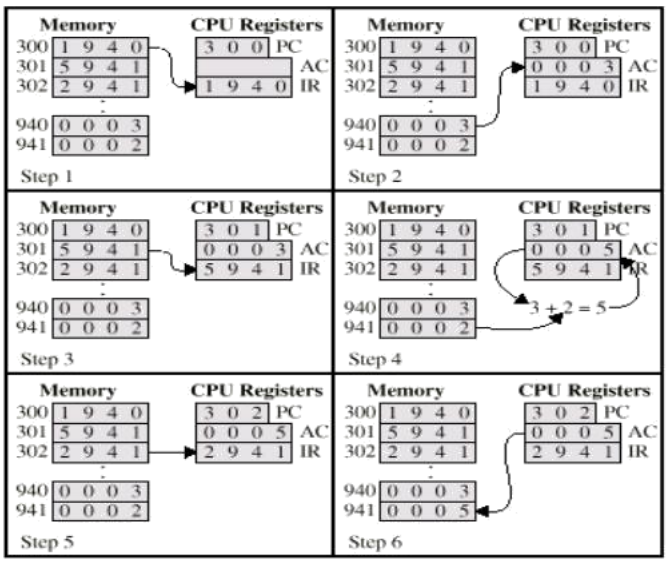
Fetch Cycle – Gidip getirme Çevrimi

- Program Sayıcısı (Program Counter (PC)) getirilecek komutun adresini tutar
- Mikroişlemci PC tarafından işaret edilen yerdeki komutu getirir
- PC arttırılır
  - — Başka türlü söylenmediği takdirde
- Getirilen komut "komut register"ına (Instruction Register (IR)) yüklenir
- İşlemci komutun kodunu çözer ve gerekli işlemleri yerine getirir

Execute Cycle – Yürütme Çevrimi

- İşlemci-bellek
  - — CPU ve ana bellek arasında veri transferi
- İşlemci-giriş/çıkış
  - — CPU ve giriş/çıkış birimi arasında veri transferi
- Veri işleme
  - — Veri üzerinde bazı aritmetik ve mantık işlemler
- Kontrol
  - — İşlemlerin sırasının değiştirilmesi
  - — örneğin jump
- Yukarıdakilerin farklı birleşimleri

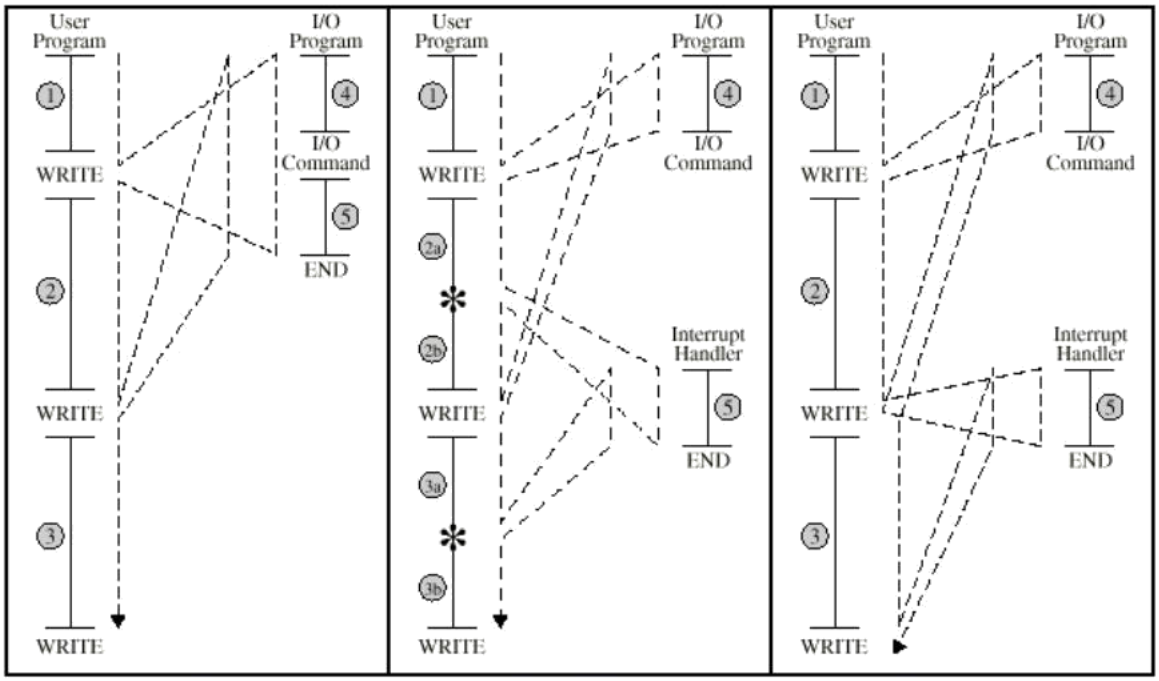
Programın çalışmasının şema ile gösterimi



KESMELER (Interrupts)

Giriş/çıkış birimi gibi bazı birimlerin normal işlem sırasını kesebildiği mekanizma (düzenek) "kesme" olarak tanımlanabilir.
Kesme'ye sebep olabilen birimler:

- Program
  - — örneğin taşma(overflow), sıfıra bölme(division by zero)
- Sistem saati
  - — Dahili işlemci saati tarafından üretilir
- Giriş/Çıkış (I/O)
  - — G/Ç kontrol edicisinden
- Donanım hatası
  - — Örneğin bellek (RAM) okunurken oluşabilen eşlik (parity) hatası

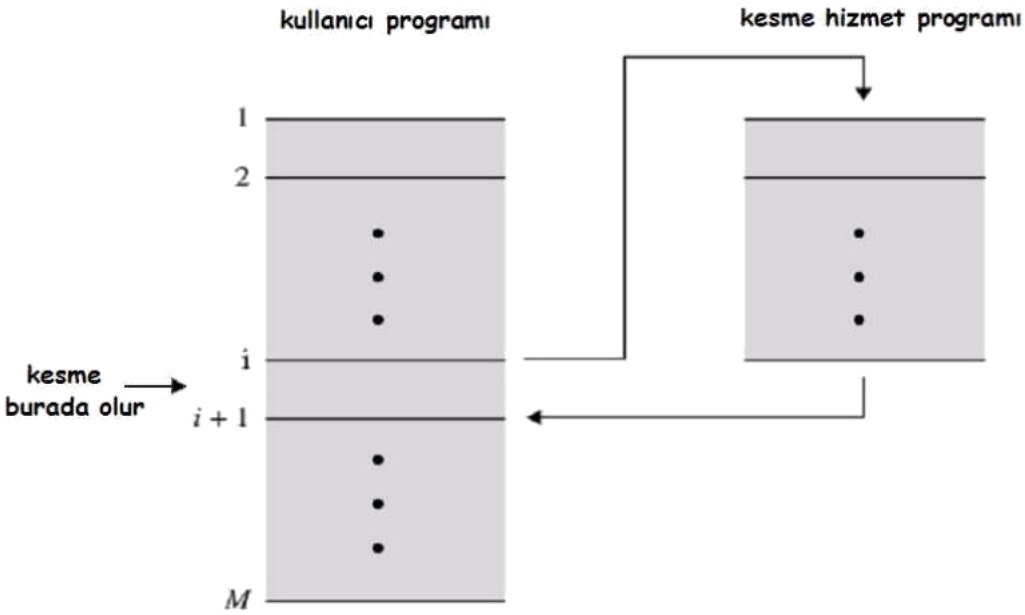


(a) No interrupts (b) Interrupts; short I/O wait (c) Interrupts; long I/O wait

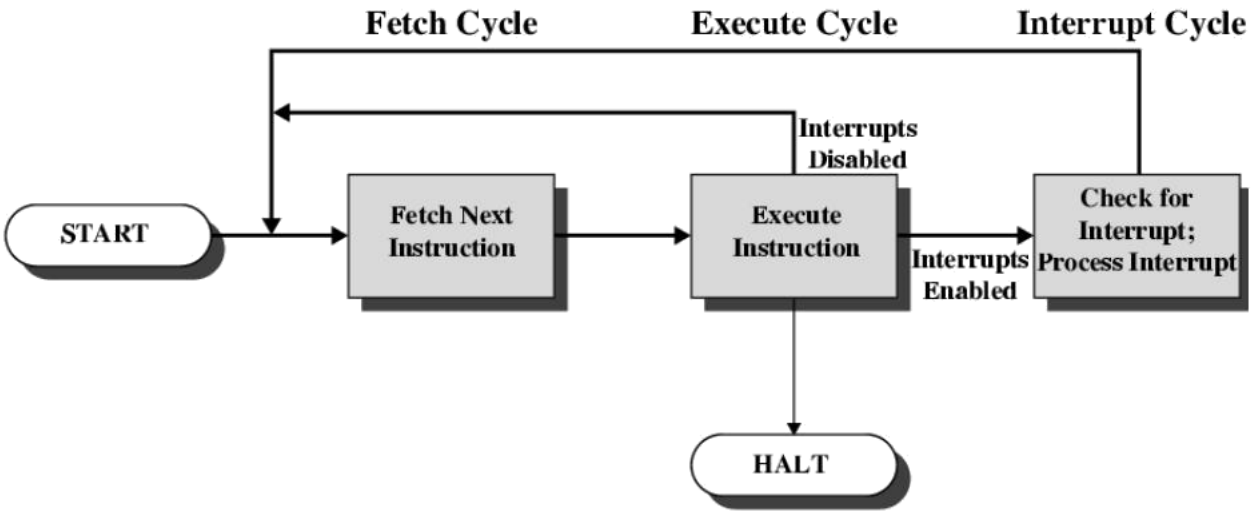
Kesme çevrimi

- Normal komut çevrimine eklenir
- İşlemci kesme isteği olup olmadığını kontrol eder
  - — Kesme sinyali tarafından belirtilir
- Eğer kesme isteği yoksa, sıradaki komutu getir
- Eğer kesme isteği varsa:
  - — Çalışmakta olan programın yürütülmesini askıya alır
  - — Genel durum bilgilerini saklar (kaydeder)
  - — Kesme hizmet programının adresini Program Sayıcısına yükler
  - — Kesme hizmet programını çalıştırır
  - — Genel durum bilgilerini tekrar yerlerine yükler ve kesilen programı çalıştırır

Kesmeler üzerinden Transfer Kontrolü



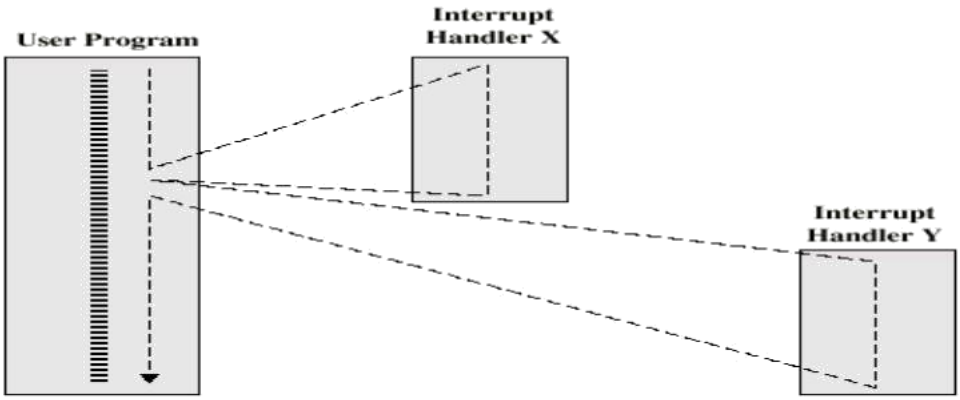
Komut Çevrimi (kesmeler dahil)



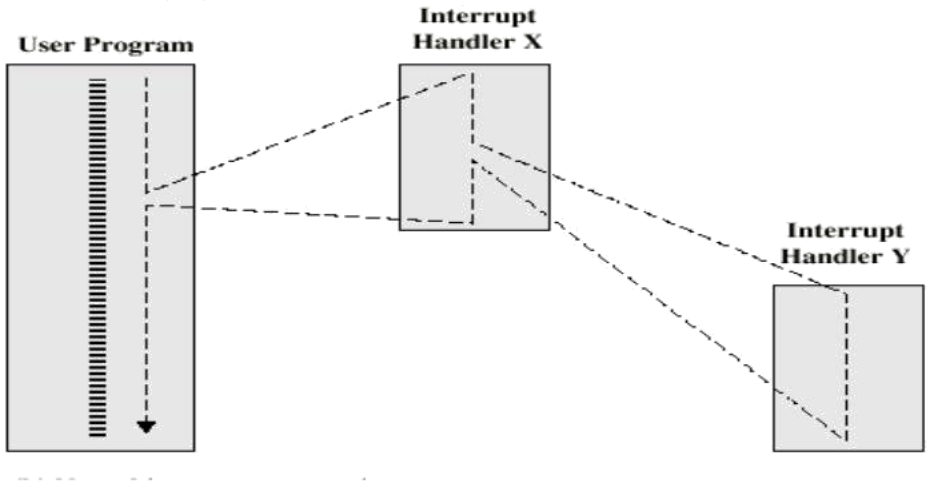
Birden fazla kesme

- Kesmelerin pasif yapılması
  - İşlemci bir kesme hizmetini gerçekleştiriyorken başka kesme isteklerini görmezden gelir
  - Kesme isteği gelmeye devam eder ve ilk kesme işlenip bitirildikten sonra diğeri tanınır
  - Kesmeler istek geliş sırasına göre yerine getirilirler
- Öncelik belirleme
  - Düşük öncelikli kesmelerin hizmet programları daha yüksek öncelikli olanlar tarafından kesilebilir
  - Yüksek öncelikli kesme işlendikten sonra işlemci önceki kesme hizmet programına döner

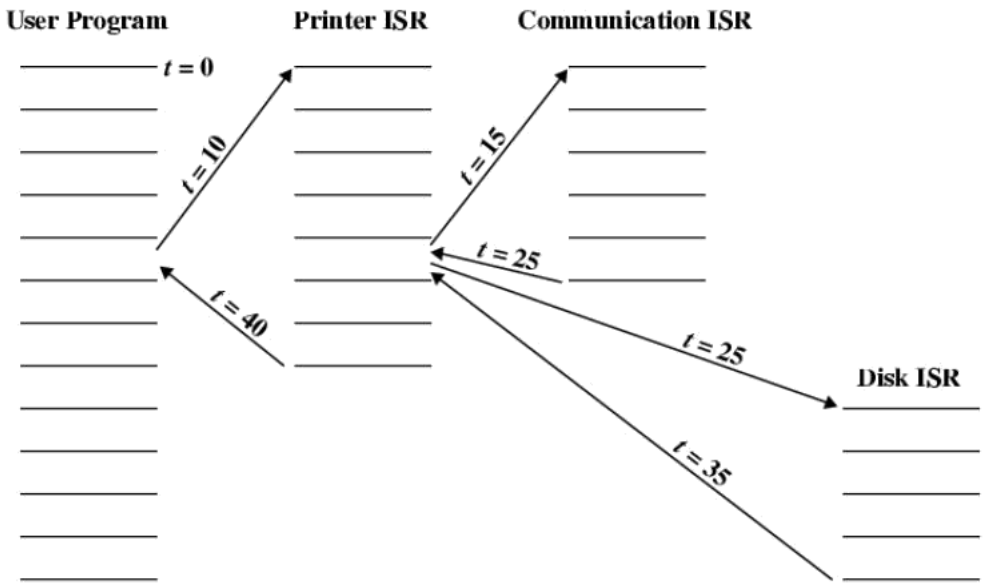
Birden fazla kesme - Sıralı



Birden fazla kesme – İç içe



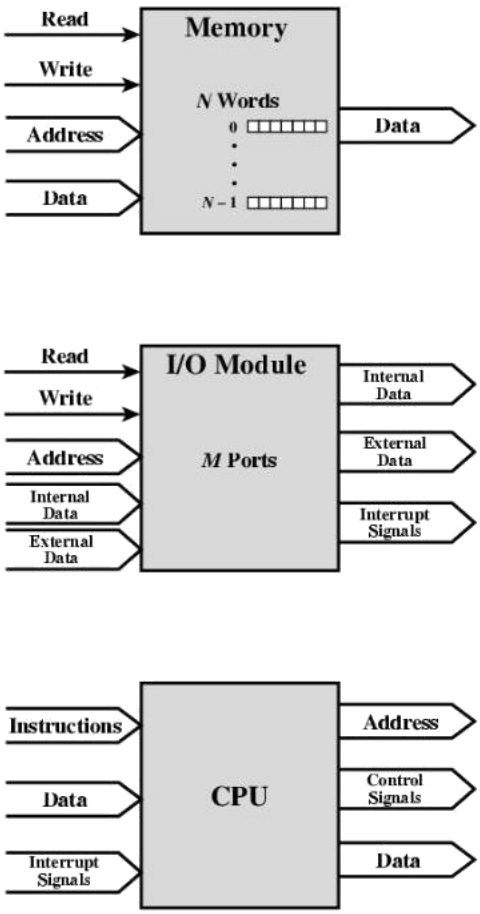
Birden fazla kesmenin zamanlama sırası



Bağlantılar

- Bütün birimler birbirine bağlanmalıdır
- Farklı tip birimler için farklı tip bağlantı
  - Bellek
  - Giriş/Çıkış
  - CPU(mikroişlemci)

Bilgisayarın bileşenleri



Bellek bağlantısı

- Veri gönderir ve alır
- Bellek satırlarının adreslerini alır
- Kontrol sinyalleri alır
  - Oku
  - Yaz
  - Zamanlama

Giriş/Çıkış bağlantısı(1)

- Bilgisayar açısından bellek ile aynıdır
- Çıkış
  - Bilgisayardan veri alır
  - Dış cihazlara veri gönderir
- Giriş
  - Dış cihazlardan veri alır
  - Bilgisayara veri gönderir

Giriş/Çıkış bağlantısı(2)

- Bilgisayardan kontrol sinyalleri alır
- Dış cihazlara kontrol sinyalleri gönderir
  - Örneğin disk
- Bilgisayardan adresleri alır
  - Örneğin dış cihazı tanımlamak için port numarası
- Kesme sinyalleri gönderir

CPU bağlantısı

- Komut ve veri okur
- İşlemlerden sonra veriyi yazar
- Diğer birimlere kontrol sinyalleri gönderir
- Kesme isteklerini alır

Ortak Yollar

- Birkaç tane olası ara bağlantı sistemi vardır
- Tek ve birden fazla ortak yol sistem yapısı en sık görülenlerdir
- örneğin Control/Address/Data bus (herbiri ayrı ayrı)
- örneğin tek ortak yol

Ortak Yol Nedir?

- İki ya da daha fazla cihazı bağlayan haberleşme yoludur
- Genellikle çok sayıda birime ulaşır
- Sıklıkla gruplandırılır
  - Bir ortak yolda çok sayıda kanal bulunur
  - örneğin 32 bit veri yolu 32 farklı tek bit kanaldan oluşur
- Enerji hatları gösterilmeyebilir

Veri Ortak Yolu (Data Bus)

- Veri taşır
  - Bu seviyede henüz "veri" ile "komut" arasında fark olmadığını hatırlayalım
- Genişlik performansın belirleyici anahtarıdır
  - 8, 16, 32, 64 bit

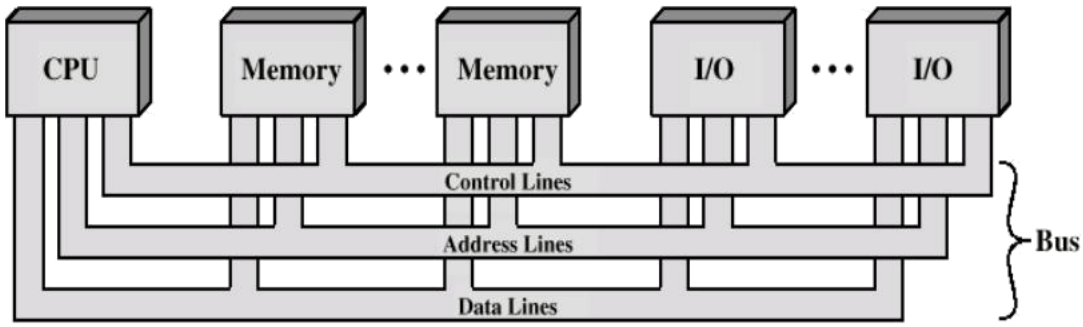
Adres Ortak Yolu (Address bus)

- Verinin kaynağını(yerini) veya hedefini(yerini) tanımlar
- Yol genişliği (yoldaki bit sayısı) sistemin maksimum bellek kapasitesini belirler
  - örneğin 8080 işlemcisinin 16 bit adres yolu 64k tane adrese erişebilir

Kontrol Ortak Yolu (Control Bus)

- Kontrol ve zamanlama bilgisi
  - Bellek oku/yaz sinyali
  - Kesme isteği
  - Saat sinyalleri

Ortak yol ara bağlantı şeması



Ortak Yollar neye benzer?

- Elektronik devre kartları üzerinde paralel hatlar
- Şerit kablolar
- Ana kart üzerinde dar ve uzun bağlantı yeri
  - örneğin PCI (Peripheral Component Interconnect)
- Bir dizi kablo

"Veri Yolları (BUS)

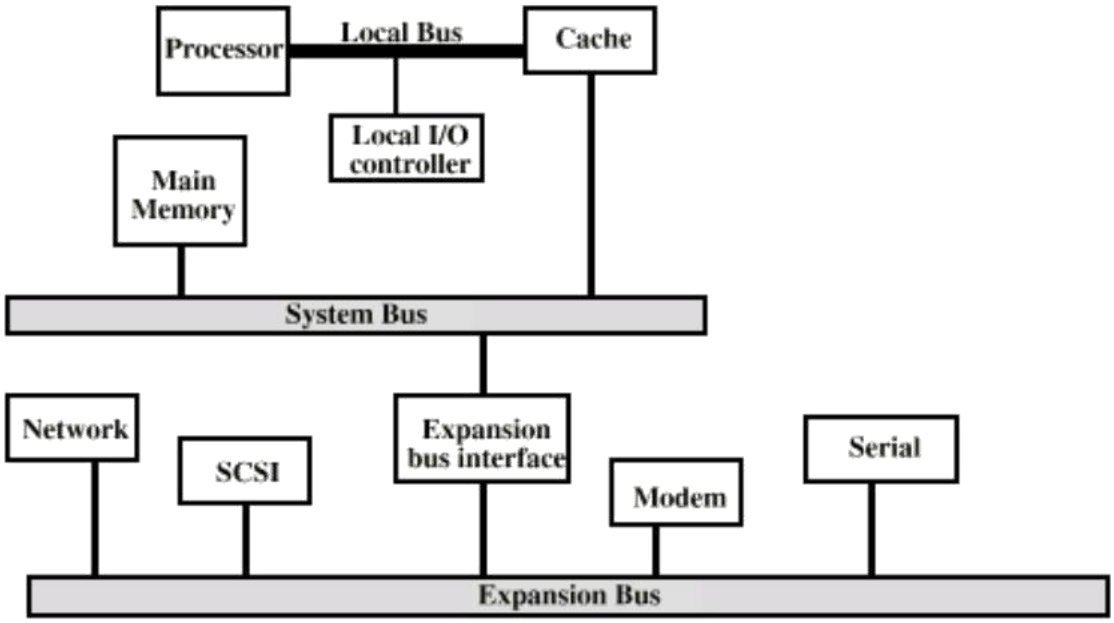
Ana kart üzerindeki bileşenlerin birbirleriyle etkileşimde bulunmasını sağlarlar. Veri yolları geliştirilme sırası ile ISA (Industry Standard Architecture), PCI (Peripheral Component Interconnect) , AGP (Advanced Graphics Port) ve PCIe (Peripheral Component Interconnect Express)'dir. Bu veri yolları, aynı zamanda bu yollar ile uyumlu çalışan ek donanım kartlarına slotlar ile bağlanabilir. Böylece veri yolunu kullanarak ek donanım birimi ile iletişim sağlanır.

Ortak yollar ile sadece veri taşınmaz. Bu yollar aynı zamanda kontrol sinyallerini ve adres bilgilerini de taşır. Kontrol sinyalleri ile donanım birimlerinin çalışmaları düzenlenir. Adres bilgileri ile donanım biriminin kullanacağı verilere ulaşım sağlanmış olur. ISA veri yolu kullanımı tamamen terkedilmiştir. Yeni üretilen ek donanım birimleri PCI veri yolunu destekleyecek şekilde üretilmektedir. Ekran kartları için kullanılan AGP veri yolu ise yerini daha hızlı veri akışı sağlayan PCIe veri yoluna bırakmaktadır."

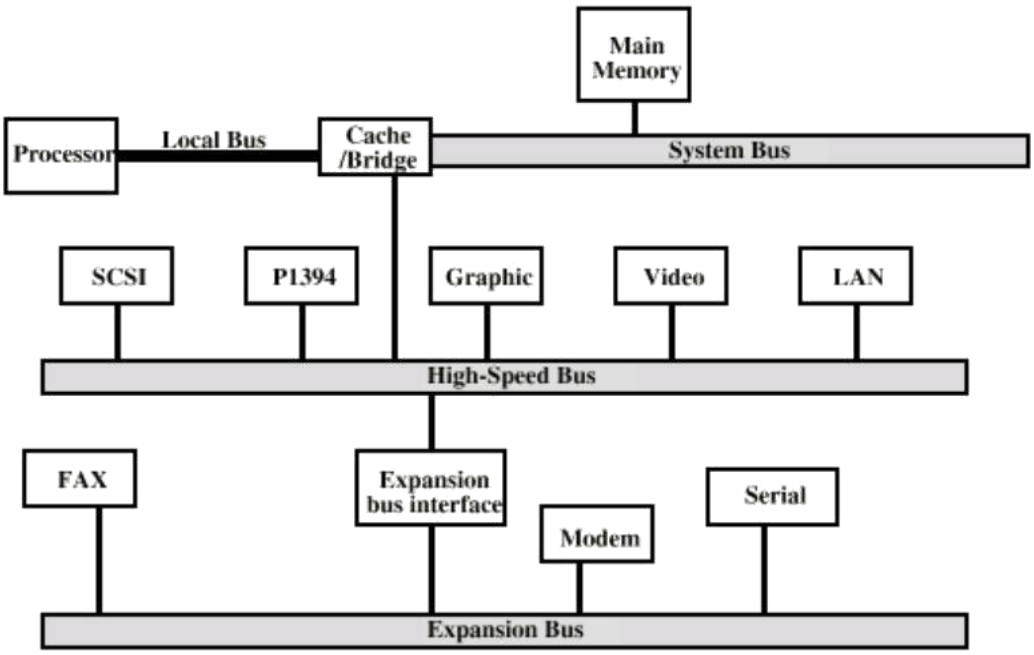
Tek ortak yol sorunları

- Tek yola bağlı çok sayıda cihaz şu sorunlara yol açar :
  - Gecikme
    - Uzun veri yolları, ortak yolun koordinasyonunu zorlaştırır ve performansı kötü etkiler
    - Toplu veri transferleri yapılırsa (ortak yola bağlı çok sayıda cihazdan iletim isteği gelirse)
    - Yolun kapasitesine yaklaşılır ve bu da gecikmelere sebep olur
- Bu sorunları aşmak için çoğu sistem çoklu ortak yol sistemi kullanır

Geleneksel (ISA) (ön bellekli--with cache)



Yüksek Performans Ortak Yolu



SCSI (Small Computer System Interface): Kendisi de yerel disk sürücülerini ve diğer dış cihazları desteklemek için kullanılan bir tür yol standartıdır.
P1394(fire-wire):Yüksek kapasiteli Giriş/Çıkış cihazlarını desteklemek için özel olarak tasarlanmış yüksek hızlı bir arabirim.

Ortak yol türleri

- Ayrı hatlar
  - Veri ve adres hatları ayrı ayrıdır
- Çoklanmış (Multiplexed)(ortak kullanım)
  - Ortak hatlar
  - 'Adres geçerli' veya 'veri geçerli' kontrol hattı
  - İyi yönü – az sayıda hat
  - Kötü yönleri
    - Daha karmaşık kontrol
    - Performansta düşme. Aynı hatları paylaştıklarından bazı olaylar paralel gerçekleşemez.

Ortak yolun paylaşımı

- Ortak yolu kullanan birden fazla birim vardır
- örneğin CPU ve DMA (Direct Memory Access-doğrudan bellek erişimi) kontrol edicisi
- Yolu aynı anda sadece bir birim kontrol edebilir (kullanabilir)
- Kontrolün kimde olacağına merkezi olarak veya yetkinin dağıtıldığı birden fazla birim olarak karar verilebilir

Merkezi karar verme

- Ortak yola erişimi tek bir donanım cihazı kontrol eder
  - Ortak Yol kontrol edicisi
  - Karar verici (hakem)
- Mikroişlemcinin parçası olabilir veya ayrı olabilir

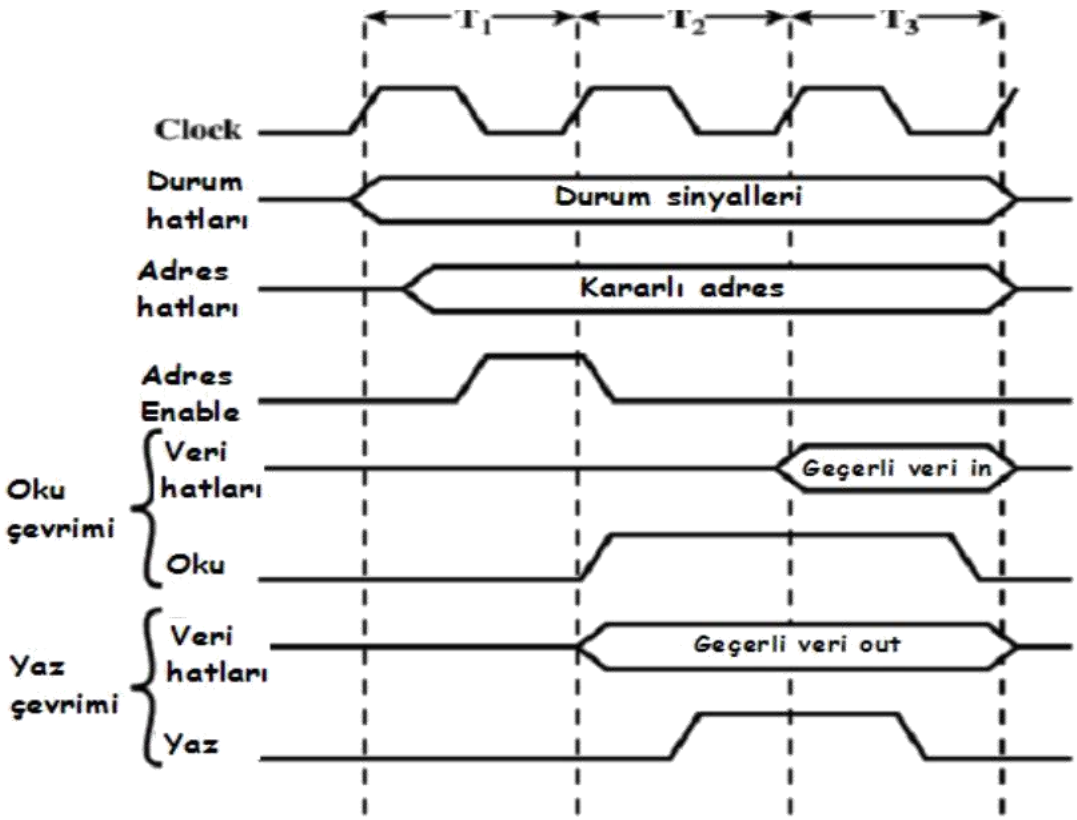
Dağıtılmış karar verme

- Her birim ortak yolu kullanma hakkı talep edebilir
- Bütün birimler üzerinde kontrol mantığı (karar verme) bulunur

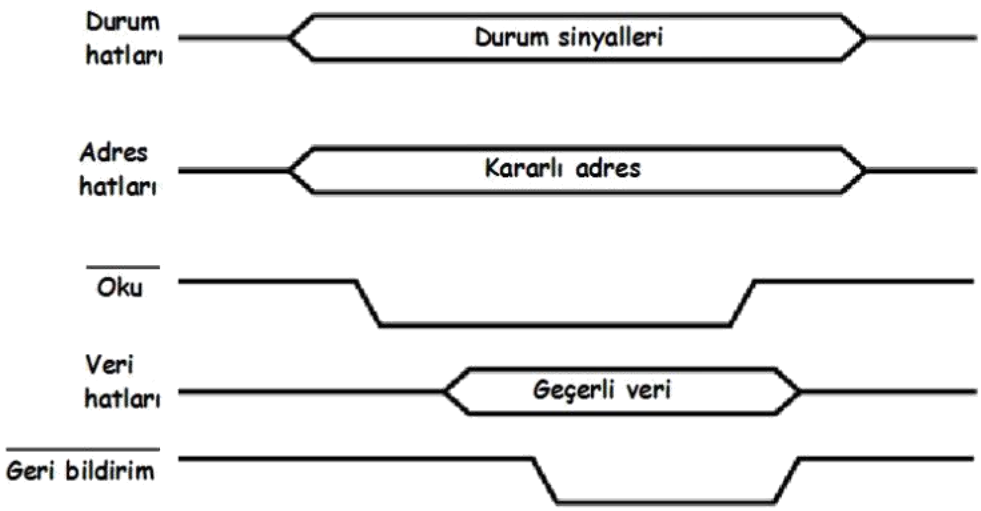
Zamanlama

- Ortak yol üzerindeki olayların (koordinasyonu) düzenlenmesi
- Eş zamanlılık (senkronize etme işlemi)
  - Olaylar saat sinyali tarafından belirlernir
  - Saat hattı kontrol yolunda bulunur
  - Bir ortak yol çevrimi tek bir periyottur
  - Bütün cihazlar saat sinyalini okuyabilir
  - Genellikle yükselen kenar ile senkronize olurlar
  - Genellikle bir olay için tek bir çevrim yeterlidir

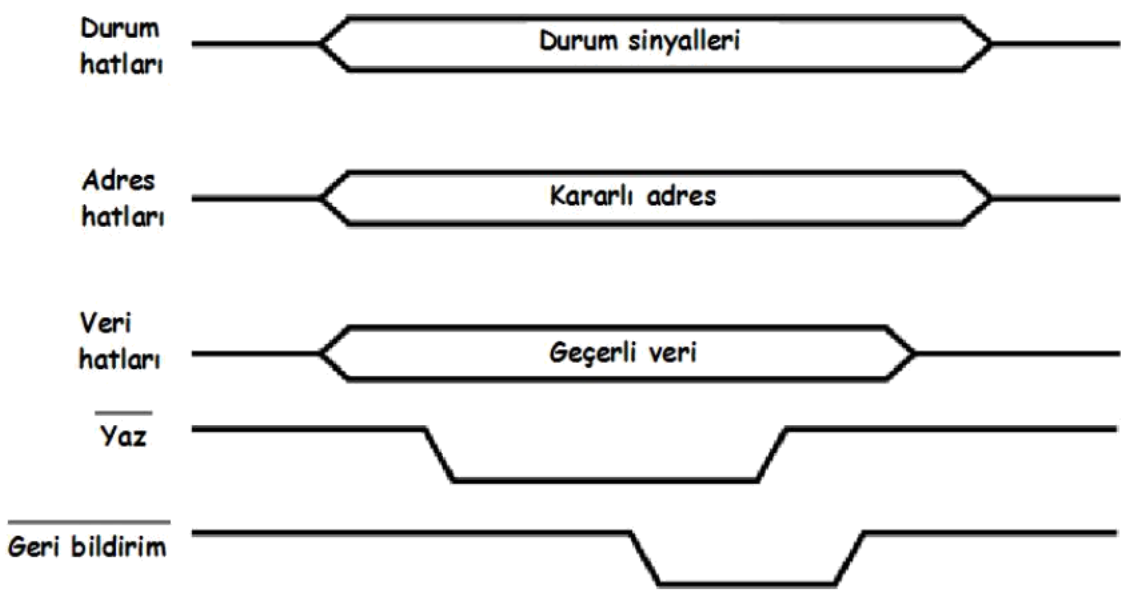
Senkron (eş zamanlı) Zamanlama Diagramı



Asenkron Zamanlama – Oku Diagramı



Asenkron Zamanlama – Yaz Diagramı



PCI Bus -- PCI Ortak Yolu

- Peripheral Component Interconnection

Dış bileşenler ara bağlantıları

- Intel buluşu
- 32 veya 64 bit (66MHz. hızında)
- 50 hat

PCI için ortak yol hatları (gerekli)

- Sistem hatları
  - Saat ve reset hatları dahil
- Adres ve Veri
  - Adres ve veri için zaman paylaşımlı 32 hat
  - Kesme ve onaylama hatları
- Arabirim Kontrolü
- Karar verme
  - Paylaşımlı değildir
  - PCI yol karar vericisine direk bağlantı
- Hata hatları

PCI için ortak yol hatları (tercihe bağlı)

- Kesme hatları
  - Paylaşımlı değil
- Ön bellek desteği

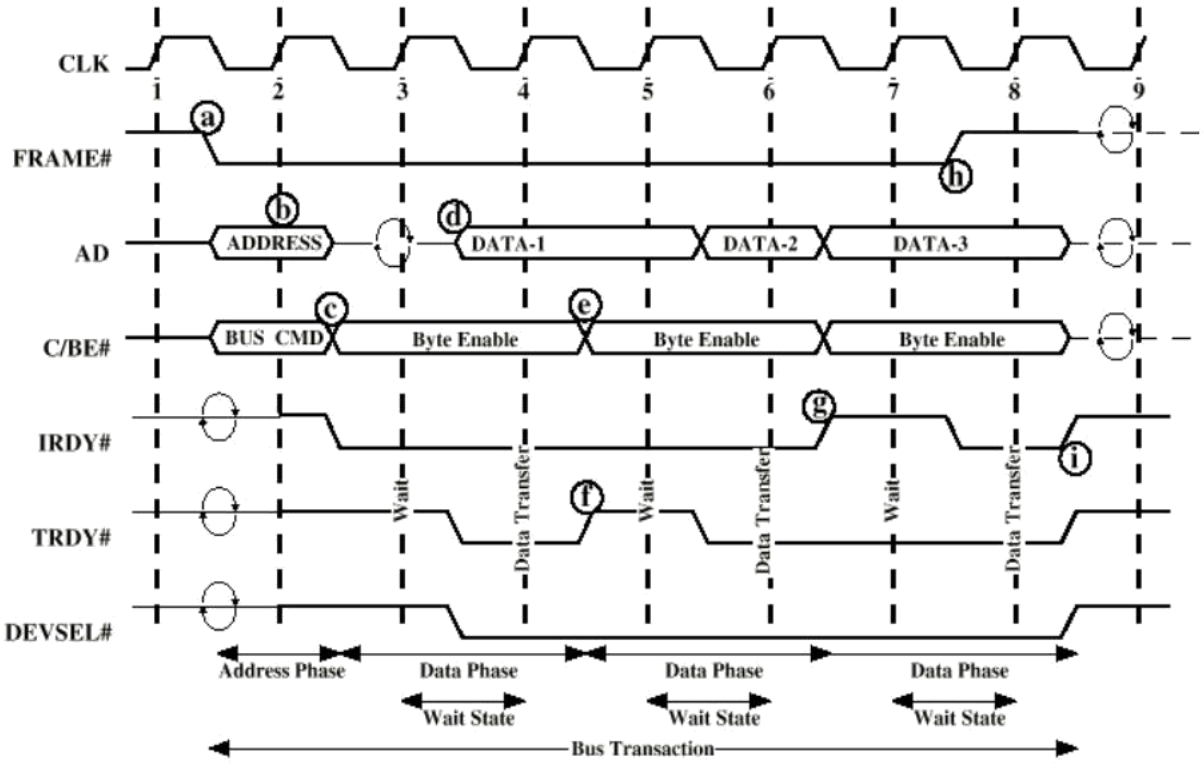
- 64-bit Bus Extension (ilave yol)
  - 32 ek hat
  - Zaman paylaşımlı
  - Cihazların 64 bit'lik transfer gerçekleştirmesi için 2 enable hattı

PCI Komutları

Gönderen ve alan taraflar arasındaki işlemler

- Gönderen ortak yolu ister
- İşlem türünün belirlenmesi
  - Yani Giriş/Çıkış Oku/Yaz
- Adres aşaması
- Bir ya da daha fazla veri aşaması

PCI Okuma İşlemi Zamanlama Diagramı



2. BÖLÜM

Ön-Bellek (Cache Memory)

2.1 BELLEKLER

Bellek sistemleri aşağıdaki özelliklerine göre sınıflandırılabilirler:

Özellikler

- Yeri
- Kapasitesi
- Transfer Birimi
- Erişim Yöntemi
- Performans
- Fiziksel tipi
- Fiziksel Özellikleri
- Organizasyonu

Yeri

- CPU (işlemci)
- Dahili (ana bellek-DRAM)
- Harici (sabit disk, cd, dvd, flash..)

Kapasitesi

- Kelime uzunluğu
  - Organizasyonun doğal birimi
- Kelime sayısı
  - Veya byte

Transfer Birimi

- Dahili
  - Genellikle veri yolu genişliği tarafından belirlenir
- Harici
  - Genellikle bir kelimeden çok daha geniş bir blok
- Adreslenebilir birim
  - Adreslenebilen en küçük bölge
  - Dahili kelime
  - Diskler üzerindeki bölümler

Erişim yöntemleri (1)

- Sıralı
  - Başından başlar ve sırasıyla okur
  - Erişim süresi verinin bulunduğu yere bağlıdır
  - Örneğin teyp bandı
- Direct-doğrudan
  - Blokların kendilerine ait adresleri vardır
  - Belirli bir bölgeye atlandıktan sonra sıralı aramayla erişilir

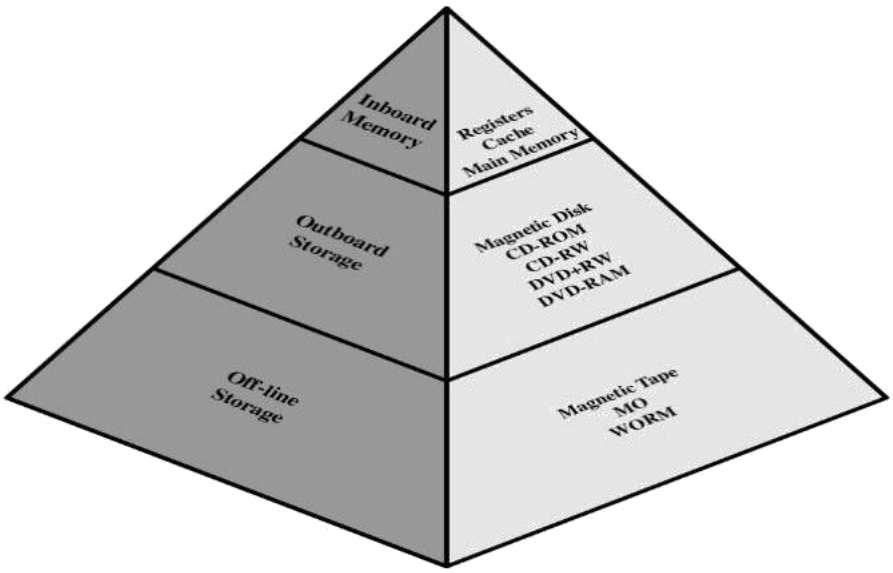
- Erişim süresi verinin bulunduğu yere ve bir önceki yere bağlıdır
- Örneğin disk

Erişim yöntemleri(2)

- Rastgele
  - Adresler bölgeleri tam olarak belirler
  - Erişim süresi erişilecek yerden veya bir önceki adresten tamamen bağımsızdır
  - Örneğin RAM veya bazı ön-bellek sistemleri
- Birleşik
  - Veri belleğin bir kısmının içeriği ile karşılaştırılarak yerleştirilir
  - Erişim süresi erişilecek yerden veya bir önceki adresten tamamen bağımsızdır
  - Örneğin ön-bellek (cache)

Bellek Hiyerarşisi

- Register'lar
  - CPU(mikroişlemci) içinde
- Dahili ya da ana bellek
  - Bir ya da daha fazla seviye ön-bellek buna dahil olabilir
    - RAM
- Harici bellek
  - Depolamak için yardımcı bellek (back-up)



Performans

- Erişim süresi
  - Adresin belirlenmesi ile geçerli verinin alınması arasındaki süre
- Bellek çevrimi süresi
  - Bir sonraki erişimden önce bellek için bir "toparlanma-iyileşme" süresi gerekebilir (işlemci ile ilgili değil yol ile ilgilidir)
  - Çevrim süresi = erişim süresi + toparlanma süresi

- Transfer hızı
  - Verinin belleğe veya bellekten iletilebilme hızı

Tipik Bellek Parametreleri

Memory Type	Technology	Size	Access Time
Cache	Semiconductor RAM	128-512 KB	10 ns
Main Memory	Semiconductor RAM	4-128 MB	50 ns
Magnetic Disk	Hard Disk	Gigabyte	10 ms, 10 MB/sec
Optical Disk	CD-ROM	Gigabyte	300 ms, 600 KB/sec
Magnetic Tape	Tape	100s MB	Sec-min., 10MB/min

Bellek türleri (Fiziksel)

- Yarıiletken
  -- RAM
- Manyetik
  -- Disk & Teyp
- Optik
  -- CD & DVD
- Diğerleri
  --Bubble
  -- Hologram

Fiziksel özellikler

- Bozulma
- Uçuculuk
- Silinebilirlik
- Harcadığı enerji

Organizasyon

- Bitlerin kelime içindeki düzeni
- Her zaman açık olmama

Sonuç

- Ne büyüklükte?
  - kapasite

- Ne kadar hızlı?
  - Zaman paradır
- Ne kadar pahalı?

Hiyerarşi sırası

- Register'lar
- L1 ön-bellek
- L2 ön-bellek
- Ana bellek
- Disk ön-bellek
- Disk
- Optik
- Bant

Hızlı mı olsun istiyorsunuz?

- Sadece statik RAM kullanan bir bilgisayar yapmak mümkündür (sonra görülecek)
- Bu çok hızlı olurdu
- Ön-belleğe ihtiyaç kalmazdı
  - Ön-belleğin ön-belleği olur mu?
- Maliyeti çok yüksek olurdu

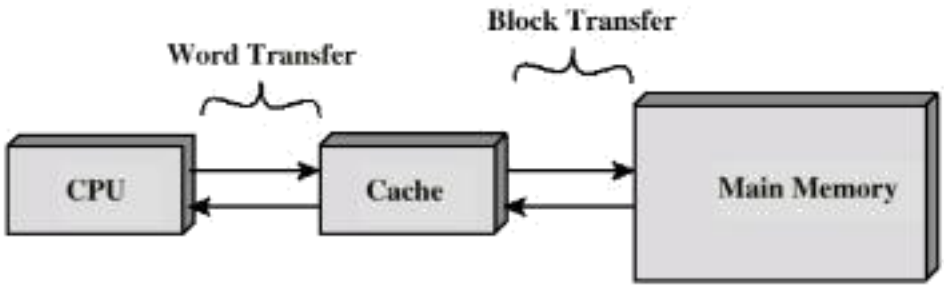
(Locality of Reference)
bulunması istenen verinin yakında olması

⇒ Programın çalıştırılması sırasında bellekte blok işlemlerine sık rastlanır
⇒ Örneğin döngüler (loop)
⇒ Böylelikle çalıştırılması gereken komutlar (bir süreliğine) aynı komutlar olacaktır
Veriyle ilgili olarak:
Örneğin array işlemlerinde gerekli veriler birbirini takip eden adreslerdedir.

2.2 ÖN-BELLEK KAVRAMI

Ön-bellek (Cache)

- Düşük kapasitede hızlı bellek
- Normal ana bellek ile CPU arasında yer alır
- Mikroişlemci çipi üzerinde de yer alabilir



Ön-bellek çalışması –özet--

- CPU bellek içeriğini okumak ister
- Önce ön-belleği kontrol eder
- Eğer istediği veri oradaysa ön-bellekten getirir (hızlı)
- Değilse, istenen bloğu ana bellekten ön-belleğe okur
- Sonra ön-bellekten CPU 'ya getirir
- Ön-bellek, herbir bölümündeki bilginin ana belleğin hangi bloğu olduğunu belirtecek etiketler kullanır

2.3 ÖN-BELLEK TASARIMI (Cache Design)

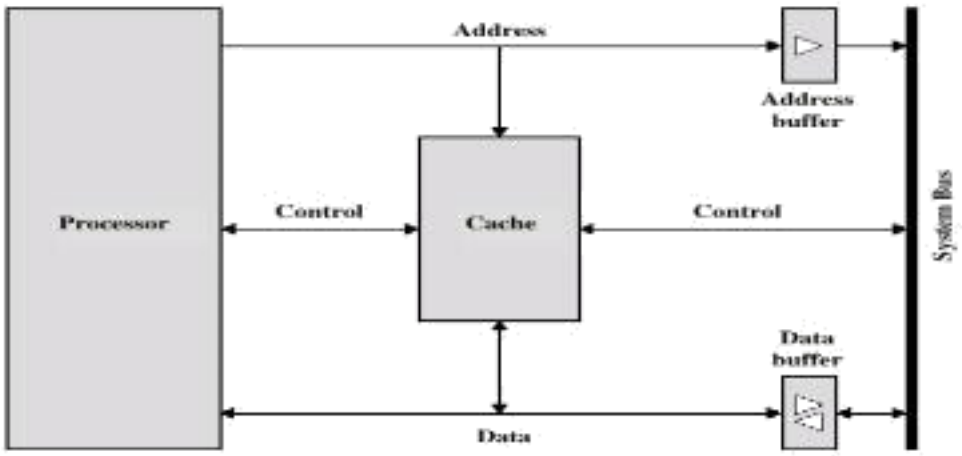
a) Büyüklük
b) Yer bulma (harita fonksiyonu-verinin yerinin belirlenmesi)
c) Yenileme algoritması
d) Yazma işlemi
e) Blok büyüklüğü (Satır Genişliği)
f) Ön-bellek sayısı

a) Büyüklük (Kapasite önemlidir)

- Maliyet
  - Daha fazla ön-bellek pahalıdır
- Hız
  - Daha fazla ön-bellek daha hızlıdır (bir yere kadar)
  - Ön-belleğin kontrolü zaman alır

Ön-bellek kapasitesinin, toplam bit maliyeti sadece ana belleğin maliyeti olacak kadar küçük, erişim süresi ise sadece ön-bellek varmış kadar büyük olması istenir.

Tipik ön-bellek organizasyonu



b) Yer Bulma (Harita fonksiyonu)

- 64kByte ön-bellek
- 4 byte'lık ön-bellek bloğu
  - yani ön-bellek 4'er byte'lık 16k (2^{14}) satırdan oluşur
- 16MByte ana bellek
  - 24 bit adres (2^{24}=16M)

1- Doğrudan Yer Bulma

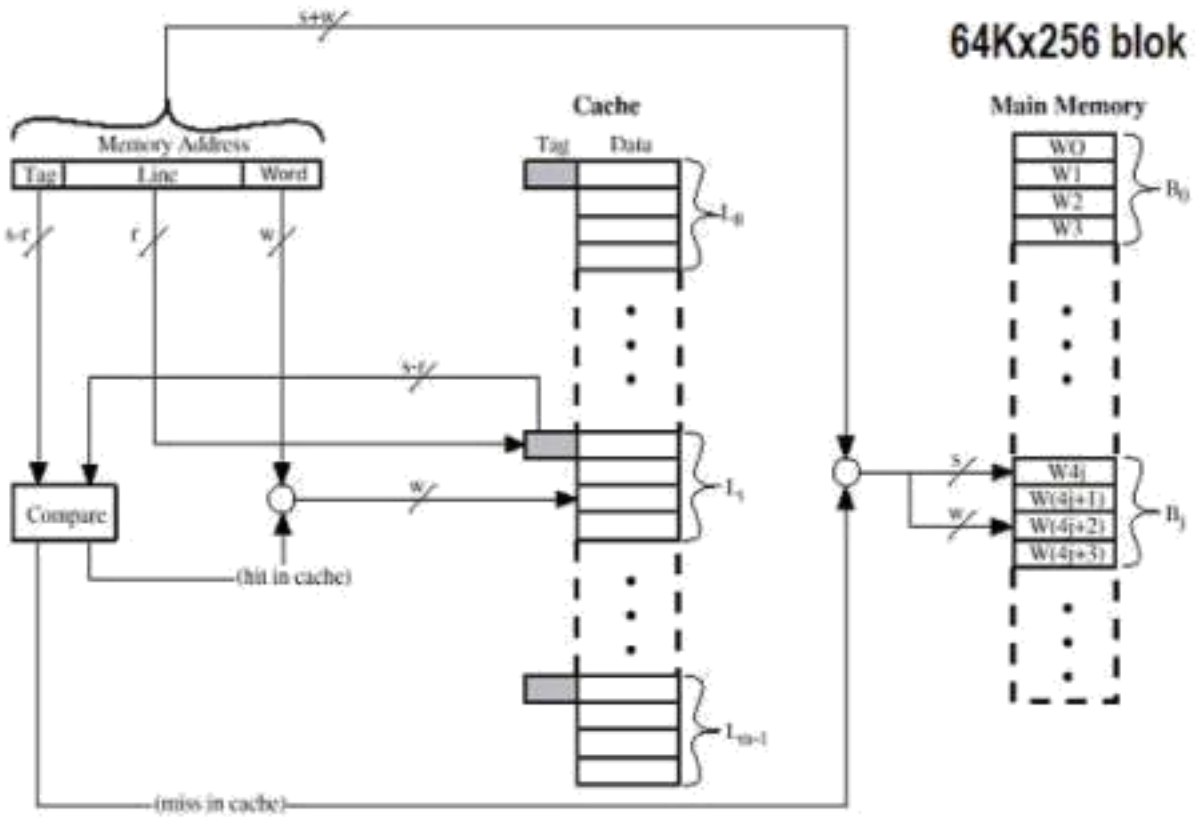
- Ana belleğin her bloğu sadece bir ön-bellek satırına işaret eder
  - Yani eğer bir blok ön-bellekte ise sadece belirli (tek) bir yerde olabilir
- Adresler iki kısım halindedir
- En az önemli (yani sağdaki) **w** tane bit tek bir kelimeyi tanımlar
- En önemli (yani soldaki) **s** ise bir bellek bloğunu belirtir
- En soldaki bitler ise ön-bellek satır alanı **r** ve s-r nin etiketi a olmak üzere ikiye ayrılır

Doğrudan yer bulma Adres yapısı

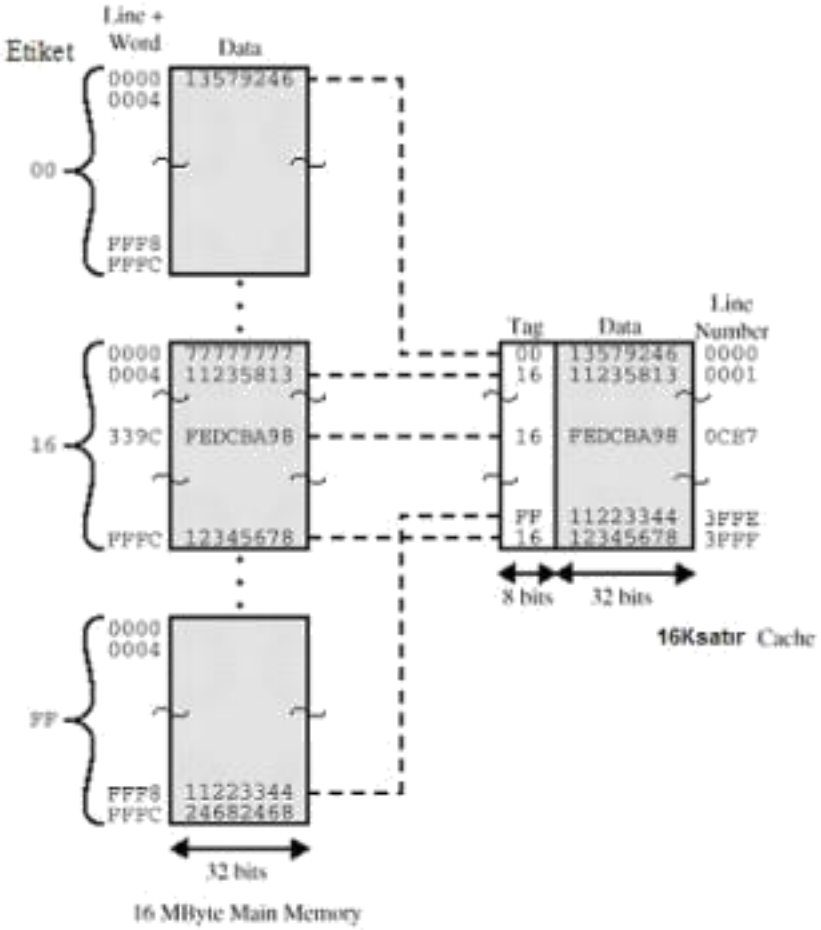
Etiket s-r	Satır veya bölge r	Word w
	14	2

- 24 bit adres
- 2 bit kelime tanımlayıcı (4 byte blok)
- 22 bit blok tanımlayıcı
  - 8 bit etiket (=22-14)
  - 14 bit satır veya bölge
- Aynı satırdaki iki bloğun etiketleri aynı olamaz
- Satır bulunarak ve etiket kontrol edilerek ön-bellek içeriği kontrol edilir

Doğrudan yer bulma ön bellek organizasyonu



Doğrudan yer bulma örneği



Doğrudan yer bulma iyi ve kötü yönleri

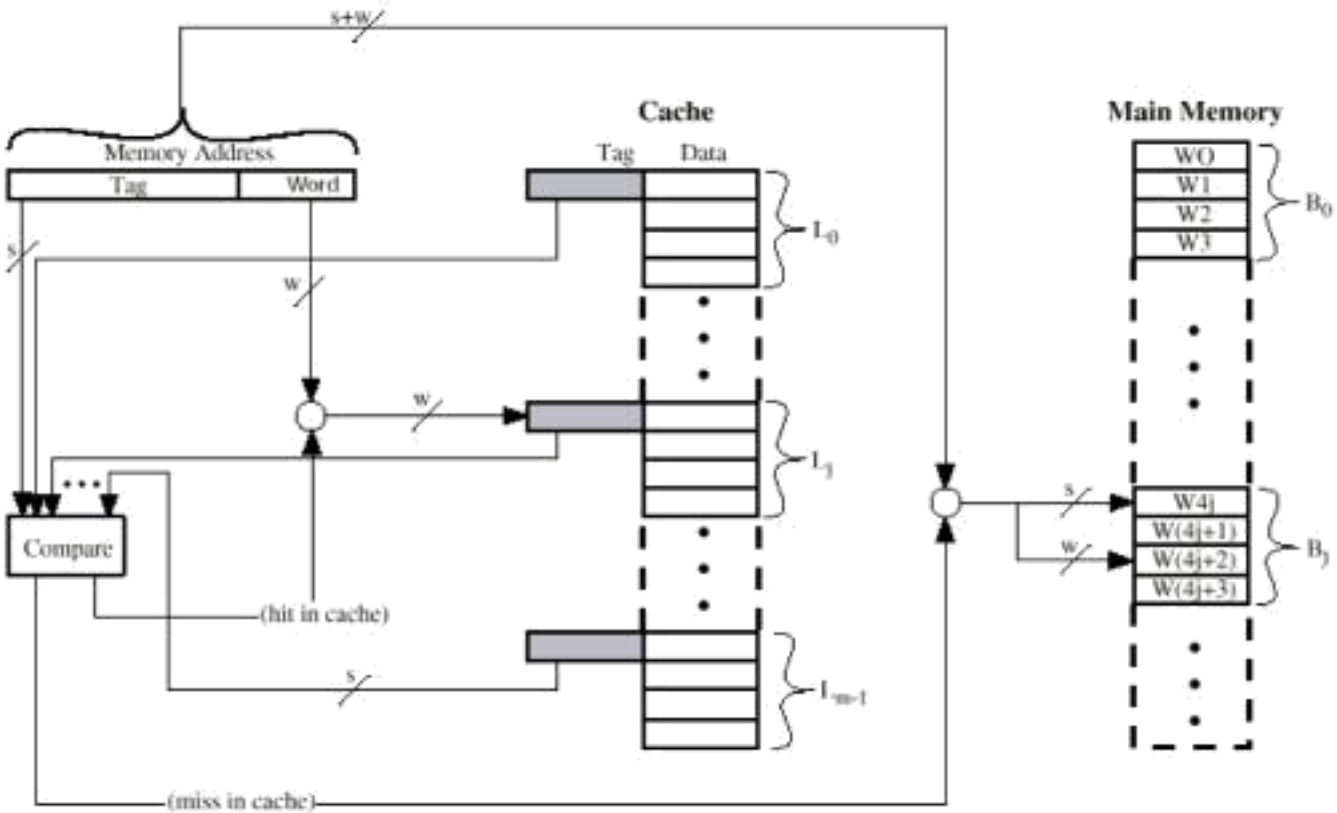
- Basit
- Ucuz
- Belirli bir blok için sabit yer
    - Eğer bir program aynı satıra işaret eden 2 bloğa tekrar tekrar erişirse, ön-bellek (miss) kaçırma olasılığı çok yüksektir

2- Birleştirilmiş Yer Bulma

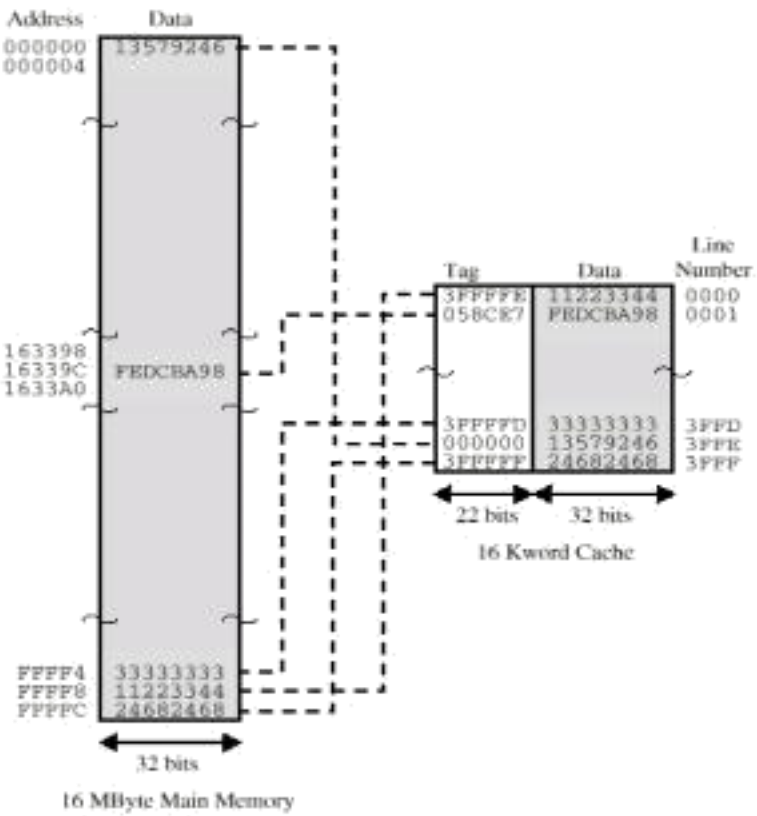
- Bir ana bellek bloğu ön-belleğin herhangi bir satırına yüklenebilir
- Bellek adresi etiket ve kelime olarak yorumlanır
- Etiket, bellek bloğunu sadece kendine özgü bir şekilde tanımlar
- Her satırın etiketi bir eşleşme olup olmadığının görülmesi için incelenir
- ön-bellek arama pahalı olmaktadır

Bu yöntemin en kötü yönü bütün etiketlerin taranması için gerekli devrenin çok karmaşık olmasıdır.

Birleştirilmiş yer bulma ön-bellek organizasyonu



Birleştirilmiş yer bulma örneği



Birleştirilmiş yer bulma Adres yapısı

Etiket 22 bit	Word 2 bit

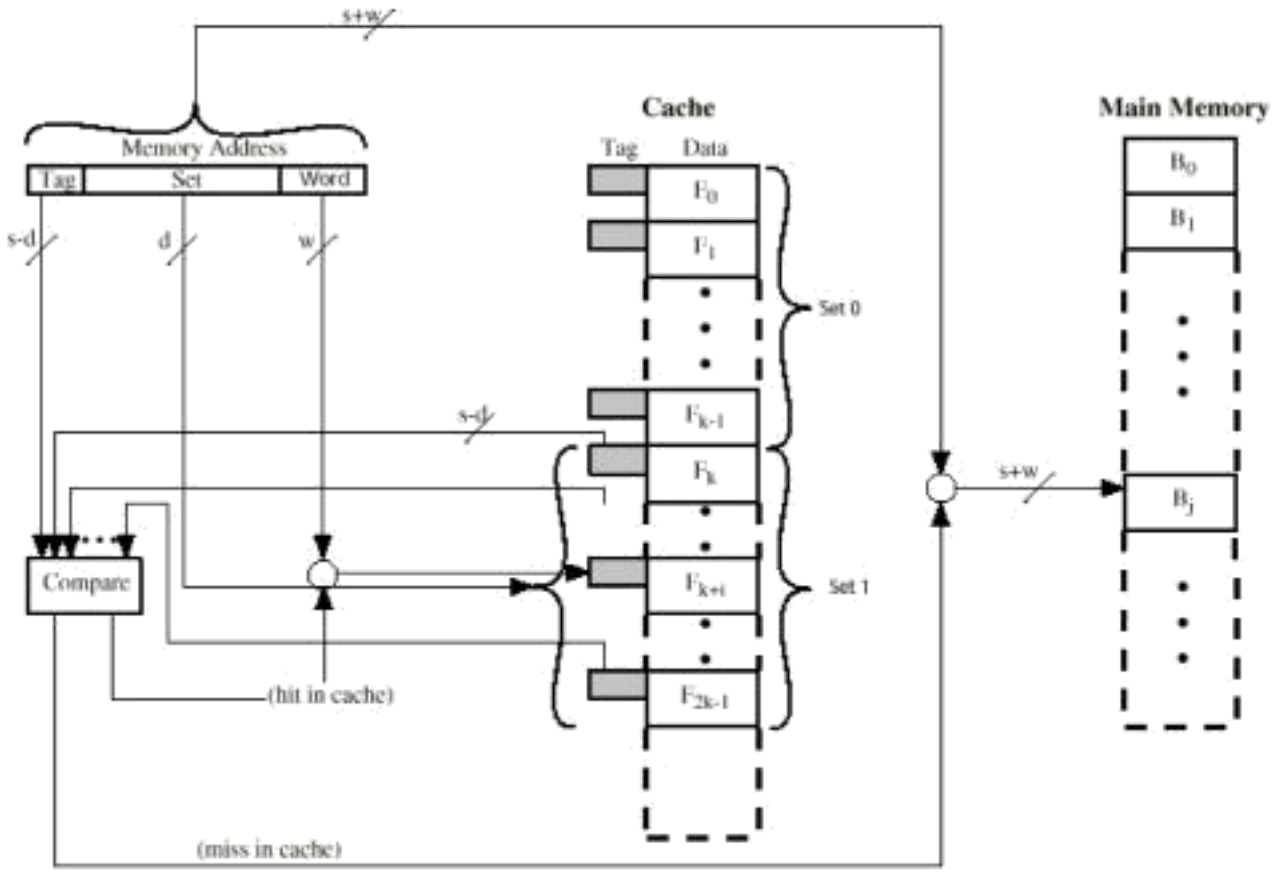
- Her 32 bitlik veri bloğu için 22 bitlik etiket saklanır
- Aranan veriyi ön-bellekte bulabilmek için etiket alanını ön-bellekteki etiket girişi ile karşılaştırır
- Adresin en sağdaki 2 biti, 32 bitlik veri bloğunun hangi 8 bitinin istendiğini tanımlar
- yani

Adres satırı	etiket	veri	ön-bellek
FFFFFC	3FFFFF	24682468	3FFF

3- Dizi Birleştirilmiş Yer Bulma

- Ön-bellek dizilere bölünmüştür
- Her dizide belli sayıda satır bulunur
- Verilen bir blok, verilen bir dizide herhangi bir satırı gösterir
  - Yani B bloğu i dizisinin herhangi bir satırında olabilir
- Yani her dizi için iki satır
  - 2 yönlü birleştirilmiş haritalama
  - Verilen bir blok, sadece bir dizideki 2 satırdan birinde olabilir

İki yönlü dizi birleştirilmiş ön-bellek Organizasyonu



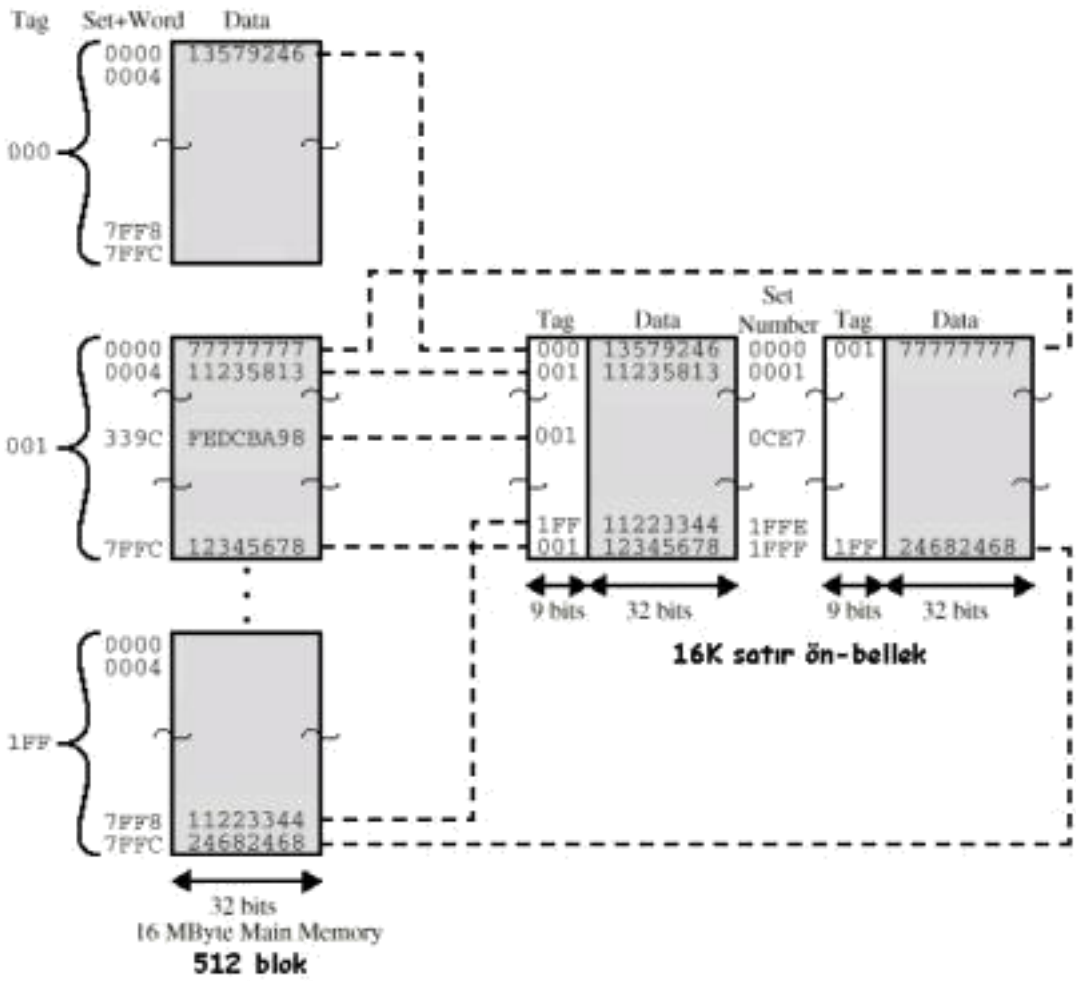
Dizi birleştirilmiş yer bulma Adres yapısı

Etiket 9 bit	Dizi 13 bit	Word 2 bit

- Bakılacak ön-bellek dizisini kararlaştırmak için dizi alanını kullanır
- Veriyi bulup bulmadığını görmek (hit-isabet) için dizi alanını karşılştırır
- yani

Adres	etiket	veri	dizi numarası
1FF 7FFC	1FF	12345678	1FFF
001 7FFC	001	11223344	1FFF

İki yönlü dizi birleştirilmiş yer bulma örneği



Bu sistem ufak bir maliyetle 4 yönlü olabilir. Performansı biraz daha arttırır. Fakat daha fazlasının önemli bir avantajı olmaz.

c) Yenileme algoritması

Yenileme algoritması (1) doğrudan yer bulma

- Başka seçenek yoktur.
- Her blok sadece bir satıra işaret eder.
- bu satır yenilenir (üzerine yeni veri yazılır).

Yenileme algoritması(2) birleştirilmiş& dizi birleştirilmiş

- Donanım ile gerçekleştirilmiş algoritma (hızlı)
- En yakın zamanda kullanılmış --Least Recently used-- (LRU)—use bit
- Yani iki yönlü dizi birleştirilmiş
  - 2 bloktan hangisi en yakın zamanda kullanılmış?
- First in first out (FIFO)—ilk giren ilk çıkar
  - Ön-bellekte en uzun süre kalan bloğu yeniler (üzerine yazar)
- Sıklıkla kullanılmayan (isabet sayacı)
  - En az isabet edilmiş (gerekmiş) olan bloğu yeniler
- Rastgele

d) Yazma İşlemi (Write Policy)

- Ana bellek güncellenmediği sürece bir ön-bellek bloğunun üzerine yazmamalıdır.
- Birden fazla CPU varsa kendilerine ait ayrı ön-bellekleri olabilir.
- Giriş/çıkış birimleri ana belleği direk olarak adresleyebilirler.

1- Hepsine Yaz (Write through)

- Bütün yazma işlemleri ön-belleğe olduğu gibi ana belleğe de gider.
- Birden fazla CPU yerel ön-belleği güncel tutabilmek için ana bellek trafiğini izleyebilir.
- Çok fazla trafiğe sebep olur.
- Yazma işlemlerini yavaşlatır.

2- Geri Yaz (Write Back)

- Güncellemeler öncelikle sadece ön-bellek üzerinde yapılır.
- Güncelleme yapılacağında ön-belleğin güncelleme biti 1 yapılır.
- Eğer bloğun üzerine yazılacaksa, ana belleğe yalnızca güncelleme biti 1 ise yazar.
- Diğer ön-bellekler, bu sırada, senkronize değildir.
- Giriş/çıkış birimleri ana belleğe ön-bellek üzerinden erişmelidirler.
- Belleğe erişimlerin %15'i yazma işlemi içindir.

e) Blok Büyüklüğü (Satır Genişliği)

Satır genişliği(line size)1

- Tasarımda düşünülecek bir diğer konudur.
- Ana bellekten bir blok veri ön belleğe getirildiğinde sadece ilgili byte değil ona bitişik başka byte'lar da getirilir.
- Satırın büyüklüğünün artması (bir noktaya kadar) iyidir. Çünkü kullanılacak veriler genelde birbirlerine yakın olanlardır. Böylelikle daha fazla faydalı veri getirilmiş olur ve isabet oranı artar.
- Ancak bir noktadan sonra isabet oranı düşer. Çünkü artık satır çok büyümüştür ve yakın zamanda getirilmiş olan bilginin kullanılma ihtimali eskiden getirilmiş olanlardan daha azdır. Yeni veri eskiden getirilmiş ve tekrar takrar gerekecek olan verinin üstüne yazılmak zorunda kalınır.

Satır genişliği(line size)2

Genişliğin artırılmasının iki önemli etkisi:

- Daha geniş satırlar o ön belleğe sığacak satır(blok) sayısını azaltır. Her yeni getirilen blok öncekilerin üzerine yazılacağından az sayıda blok olması, getirildikten kısa bir süre sonra üstüne yazılarak kaybedilmesi ile sonuçlanır.
- Satır genişliği arttıkça her bir kelime gerçekte istenen kelimeden uzakta olur ve yakın gelecekte ihtiyaç duyulma ihtimalini azaltır.

Satır genişliği(line size)3

- Satır genişliği ile isabet oranı arasındaki bağıntı çok karmaşıktır ve bu yüzden en iyi denebilecek bir rakam yoktur.
- 8 ile 32 byte arasındaki genişlikler sıklıkla kullanılır.
- Yüksek performanslı sistemlerde 64 ve 128 byte en sık rastlanılanlardır.

f) Ön-bellek Sayısı

1- Kaç tane ön bellek?1

- Ön bellekler ilk kullanıldıklarında tektiler.
- Şimdi birden fazla seviye yaygındır.
- Entegrelerin yoğunluğu arttıkça ön belleğin mikroişlemci çipine yerleştirilmesi mümkün oldu.
- Çip üzerindeki ön belleğin varlığına rağmen çip dışında fazladan bir ön belleğe hala ihtiyaç var mıdır?

Kaç tane ön bellek?2

- Cevap: evet
- Çip üzerindeki L1 ve dışındaki L'2 olmak üzere:
- L2 yoksa ve işlemci L1'de olmayan bir veriyi istiyorsa sistem yolu üzerinde bulunan DRAM veya ROM'a başvurmak zorunda olacak. Yolun yavaş oluşu ve bu belleklerin yavaş oluşu performansı düştürecektir.

- Oysa L2'nin olması durumunda, verinin L1'de bulunmamasının olumsuzluğu telafi edilebilir. Eğer Statik RAM yeterince hızlı ise hiç "wait state" eklenmeden veriye erişilebilir.

Kaç tane ön bellek?3

- L2 ile işlemcinin veri alışverişi çoğunlukla sistem yolu üzerinden değildir. İkisi arasında ayrı bir yol kullanılır. Bu da sistem yolunun yükünü azaltır.
- İşlemci birimlerinin gittikçe küçülmesi ile birlikte bazı işlemcilerde L2 de çip üzerine yerleştirilmiştir.
- L2 ön belleğin kullanımının sağladığı fayda hem L1 hem de L2'nin isabet oranlarına bağlıdır.
- Genel olarak L2'nin kullanımının performansı arttırdığı söylenebilir.
- Ancak birden fazla seviye ön bellek kullanımı tasarımla ilgili konuları karmaşıklaştırır.(kapasitesi, yerleştirme algoritması, yazma kuralları gibi)

2- Birleşik mi ayrı ön bellek mi?1

Veri ve komutlar için tek bir birleşik ön belleğin avantajları:

1. Belirli bir kapasite için birleşik ön bellek daha yüksek isabet oranlarına sahiptir. Çünkü komut ya da veri getirme oranlarını kendiliğinden dengeler. Yani o sırada yapılan iş daha çok komutların getirilmesine sebep oluyorsa ön bellek onlarla doldurulur.
2. Sadece bir önbelleğin tasarlanması ve gerçekleştirilmesi gerekir. Bu da daha kolaydır.

Birleşik mi ayrı ön bellek mi?2

Birleşik ön belleğin avantajlarına rağmen eğilim ayrı olanlara doğrudur.
Komutların paralel çalışması ve çalışma sırası tahmin edilen komutların önceden alınması, ayrı ön bellekleri avantajlı duruma getirir.
Ayrı ön bellek, komutu getirme ve kod çözme birimleri ile çalıştırma biriminin çatışmasını önler. Bu özellikle iş hattı kullanan işlemcilerde önemlidir.

3- Pentium IV ön bellek organizasyonu

- 80386'nın işlemci üzerinde önbelleği yoktur.
- 80486'nın işlemci üzerinde, 8 KByte kapasitede, 16 byte satır büyüklüğünde ve 4 yönlü birleştirilmiş dizi yapısında önbelleği vardır.
- Bütün Pentium'larda çip üzerinde veri ve komut için ayrı olmak üzere L1 seviye ön belleği vardır.
- Pentium 4'ün çip üzerinde 8 KByte kapasitede, 64 byte satır büyüklüğünde ve 4 yönlü birleştirilmiş dizi yapısında, geri yazma kurallı bir veri önbelleği vardır.

(Pentium 4'ün çip üzerindeki komut önbelleği yine 8 KByte'dır ve daha sonra incelenecektir.)

- Pentium 4'ün ayrıca hem veri hem de komut L1'lerini besleyen 256 KByte kapasitede, 128 byte satır büyüklüğünde ve 8 yönlü birleştirilmiş dizi yapısında L2 ön belleği bulunur.

3. BÖLÜM

Dahili Bellek (Internal Memory)

ÖZET:

1. Yarı iletken belleklerin en temel iki türü daha hızlı, daha pahalı ve daha az yoğun olan ve ön bellek yapımında kullanılan Statik RAM (SRAM) ve ana bellek olarak kullanılan Dinamik RAM'dir(DRAM). Her ikisi de rastgele erişimlidirler.

2. Hata düzeltme teknikleri bellek sistemlerinde yaygın olarak kullanılırlar. Veri bitlerinin değerine bağlı olarak fazladan bit eklenmesiyle bir hata düzeltme kodu oluşturulur. Eğer bir bit hatası olursa bu kod yardımıyla tespit edilir ve genellikle düzeltilir.

3. Nispeten düşük hızda çalışan DRAM'i biraz daha iyileştirebilmek için birkaç ileri DRAM organizasyonları geliştirilmiştir. En yaygın ikisi Senkron DRAM ve RamBus DRAM'dir. Bunların ikisi de veriyi bloklar halinde taşıyabilmek için sistem saati kullanımı ile ilgilidir.

3.1 YARI İLETKEN ANA BELLEK

Yarı iletken Bellek Türleri

<table>
<tr><th>Bellek Türü</th><th>Sınıfı</th><th>Silinebilirlik</th><th>Yazma Düzeneği</th><th>Uçuculuk</th></tr>
<tr><td>Random-access memory (RAM)</td><td>Read-write memory</td><td>Elektrik ile, byte seviyesinde</td><td>Elektrik ile</td><td>Uçucu</td></tr>
<tr><td>Read-only memory (ROM)</td><td rowspan="2">Read-only memory
Sadece okunabilir</td><td rowspan="2">Mümkün değil</td><td>Maske yöntemi</td><td rowspan="5">Uçucu Değil</td></tr>
<tr><td>Programmable ROM (PROM)</td><td rowspan="4">Elektrik ile</td></tr>
<tr><td>Erasable PROM (EPROM)</td><td rowspan="3">Read-mostly memory
Çoğunlukla okunur</td><td>Mor-ötesi ışınlar ile, Çip seviyesinde</td></tr>
<tr><td>Electrically Erasable PROM (EEPROM)</td><td>Elektrik ile, byte seviyesinde</td></tr>
<tr><td>Flash memory</td><td>Elektrik ile, Blok seviyesinde</td></tr>
</table>

Yarıiletken Bellek

- RAM
  - Bütün yarıiletken bellekler rastgele erişimli olduğundan yanlış isimlendirilmiştir
  - Okunabilir/yazılabilir
  - Uçucudur
  - Geçici depolama sağlar
  - Statik veya dinamik olabilir

Bellek Hücresi

Dinamik RAM

- Bitler kondansatör üzerindeki yük olarak kaydedilir
- Bu yük kendi kendine boşalır
- Enerji varken bile verinin yenilenmesi gerekir
- Yapısı daha basittir
- Bir bit daha az yer kaplar
- Daha ucuzdur
- Yenileme devresine ihtiyaç duyar
- Daha yavaştır
- Ana bellekler bu şekildedir
- Aslında analogdur
    - Verinin değerini yükün seviyesi belirler

Statik RAM

- Bitler açık/kapalı anahtarlar olarak depolanır
- Yükün boşalması durumu yoktur
- Enerji varken verinin tazelenmesi gerekmez
- Daha karmaşık yapıya sahiptir
- Bir bitin depolanması için daha fazla yer gerekir
- Daha pahalıdır
- Tazeleme devresine ihtiyacı yoktur
- Daha hızlıdır
- Ön-bellekler bu şekildedir
- Dijitaldir
    - Flip-floplardan oluşur

SRAM ve DRAM karşılaştırılması

- Her ikisi de uçucudur
    - Verinin korunması için enerji gerekir
- Dinamik hücre
    - İmali daha basit, daha küçük
    - Daha yoğun (birim alanda daha fazla bit)
    - Daha ucuz
    - Tazeleme gerekir
    - Daha büyük kapasitede belle birimleri
- Statik hücre
    - Daha hızlı
    - Ön-bellek içinde kullanılır

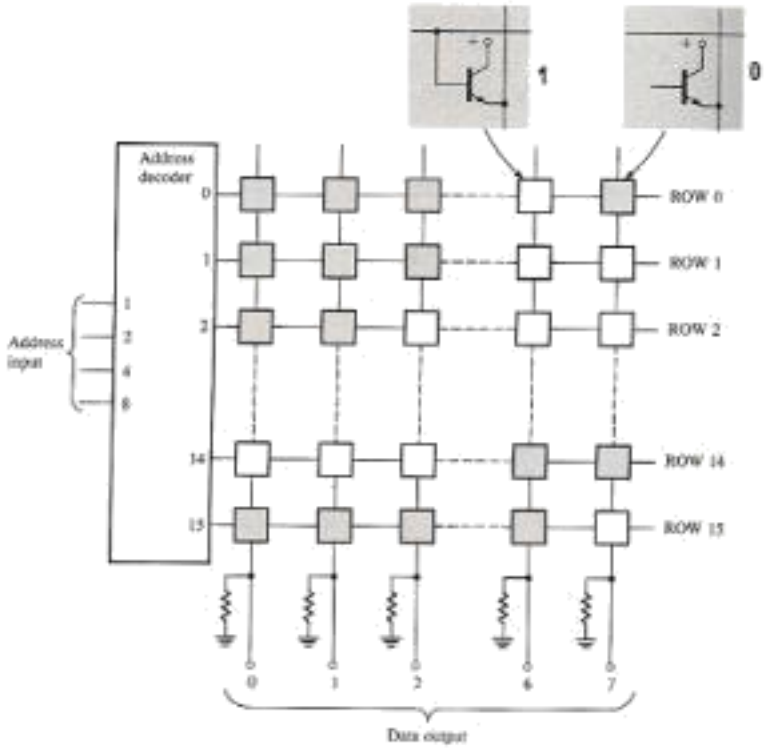
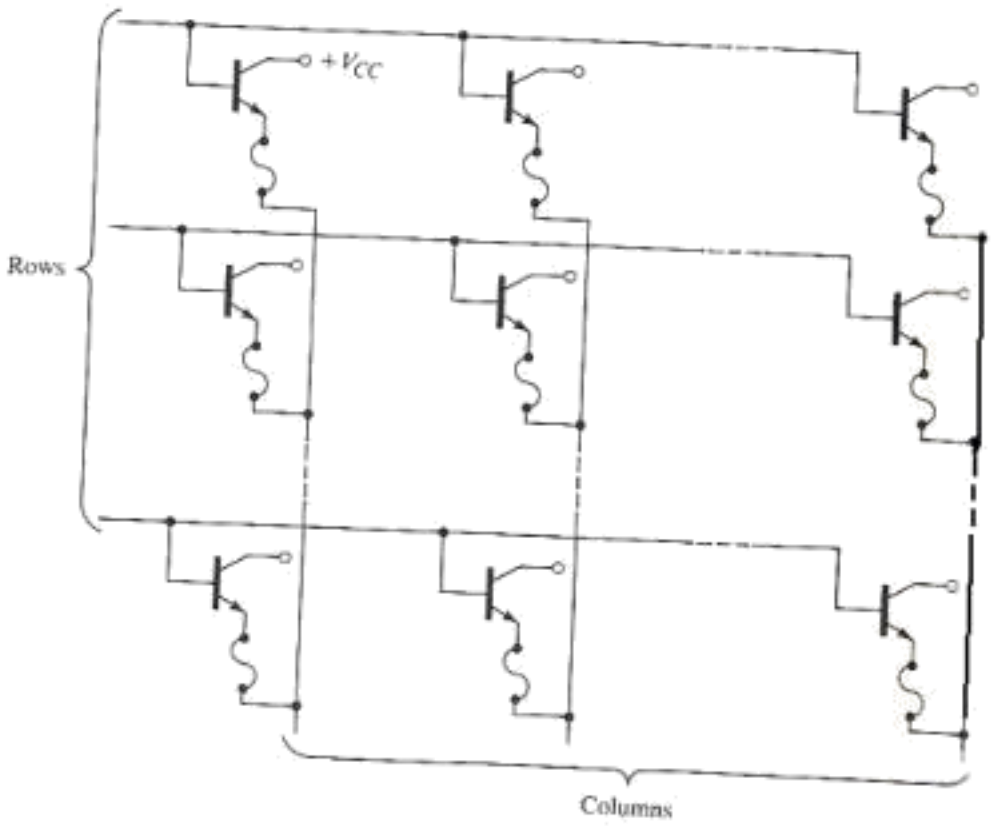
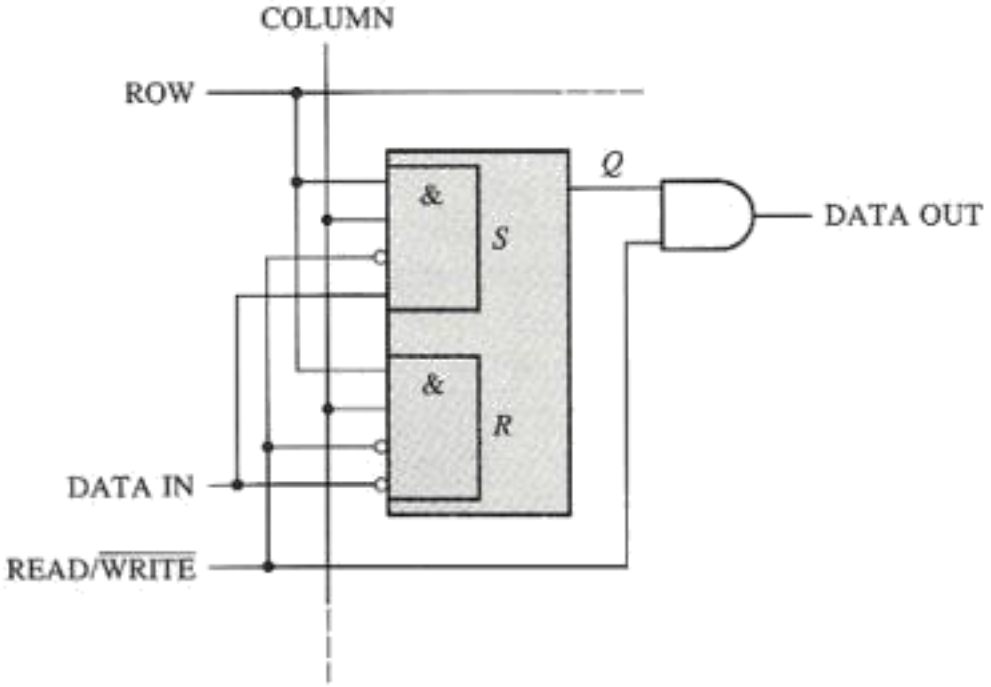


FIGURE 10–10 *A 16X8-bit ROM array.*



MASK ROM

Bir Bit RAM



Sadece Okunabilir Bellek (ROM)

- Kalıcı depolama
  - Uçucu değildir
- Mikroprogramlama (sonra görülecek)
- Kütüphane altprogramları
- Sistem programları (BIOS)
- Fonksiyon tabloları

ROM Türleri

- Üretim sırasında yazılır
  - Az sayıda çalıştırma için veya az sayıda çip için çok pahalı
- Programlanabilir (bir defa)
  - PROM
  - Programlamak için özel cihazlara gerek vardır
- Çoğunlukla okunur
  - Silinebilir Programlanabilir (EPROM)
    - Mor ötesi (UV) ışınlarla silinebilir(yaklaşık 20 dak. sürer)
  - Electrikle silinebilir (EEPROM)
    - Yazmak okumaktan çok daha fazla zaman alır(her byte için birkaç yüz mikrosaniye)
  - Flash bellek
    - Elektrik ile silinebilir
    - İlk olarak 1980'lerin ortalarında tanıştırıldı
    - Çok hızlıdır
    - Her bit için bir transistör kullanılır(yoğunluk çok yüksek)

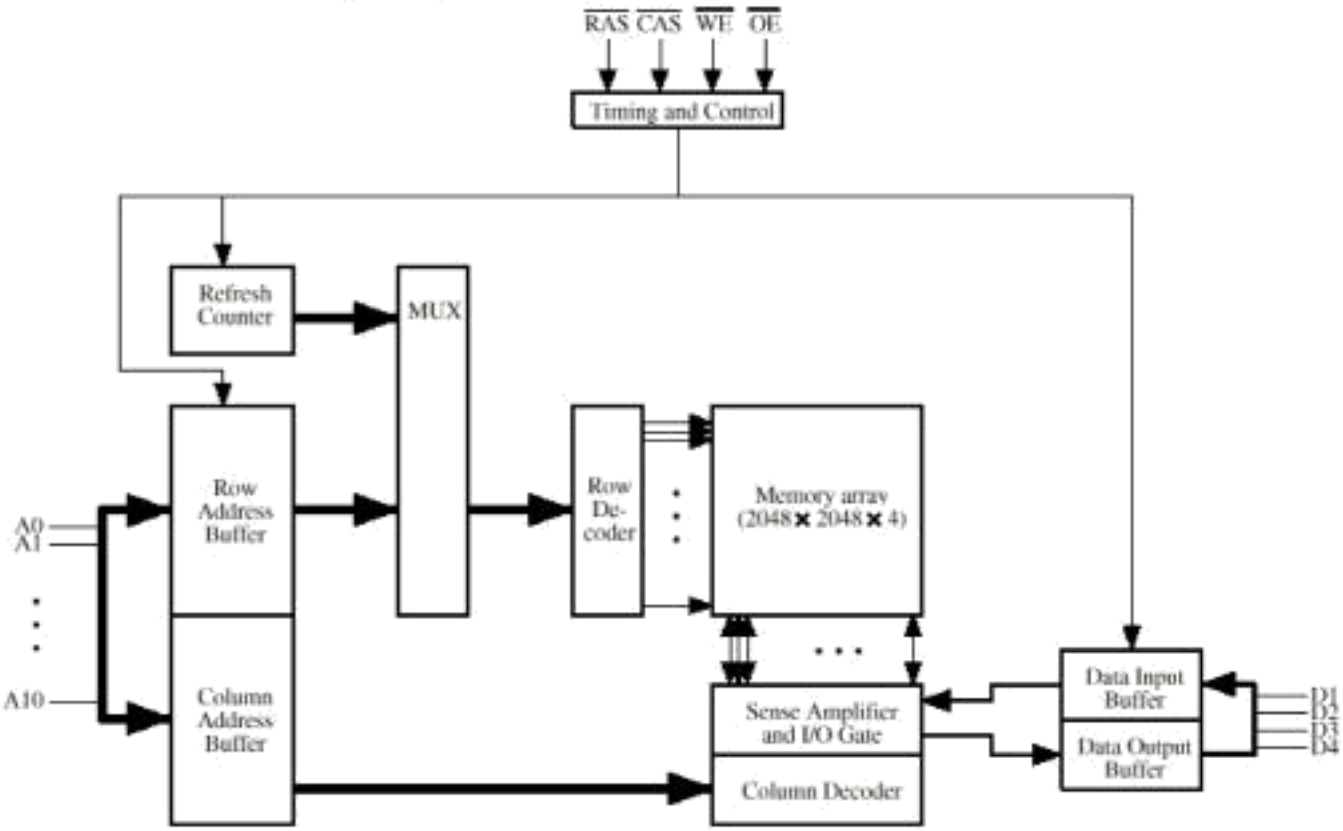
Ayrıntılarıyla organizasyon

- 16Mbit'lik bir bellek entegresi(çip) 1M tane 16 bit'lik kelime olarak düzenlenebilir
- Her çipte 1 bit sistemi 16 tane 1Mbit'lik çip ile düzenlenebilir
- 16Mbit'ik çip 2048 x 2048 x 4bit dizisi olarak düzenlenebilir
  - Entegre üzerindeki adres bacak sayısını (pin) azaltır
    - Çoklanmış (Multiplex) satır ve sütun adresleri
    - Adreslemek için 11 pin (2^{11}=2048)
    - Bir fazla pin eklenerek değer aralığı iki katına çıkarılabilir o halde kapasite 4 katına çıkar

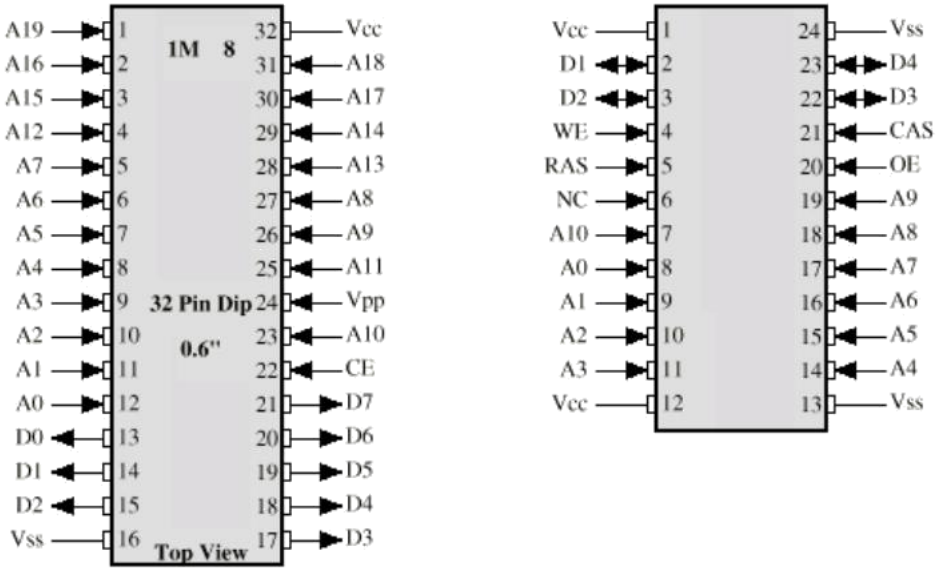
Refreshing(tazeleme)

- Tazeleme devresi çip üzerindedir
- Tazeleme sırasında çipi devre dışı bırakır
- Satırlar üzerinden sayılır
- Okunur ve tekrar üzerine yazılır
- Zaman alır
- Performansı farkedilir derecede düştürür

Tipik bir 16 Mb DRAM (4M x 4)

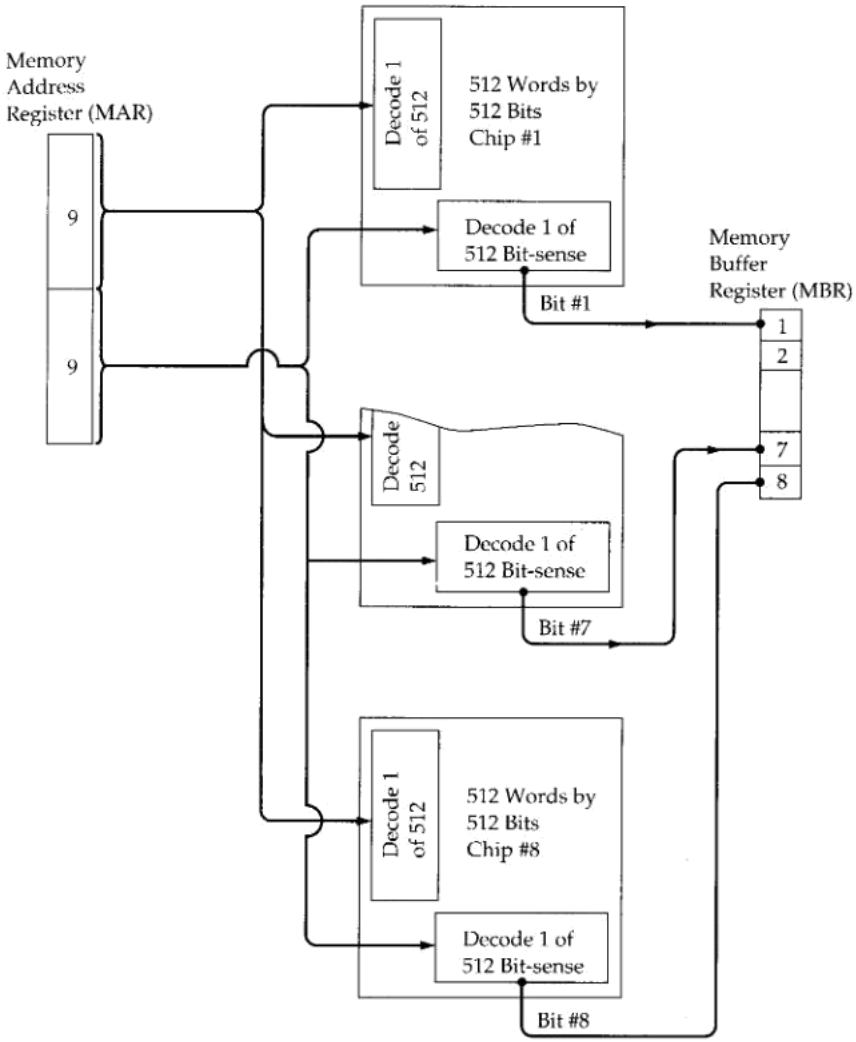


Çip üzerindeki bacak (pin) bağlantıları

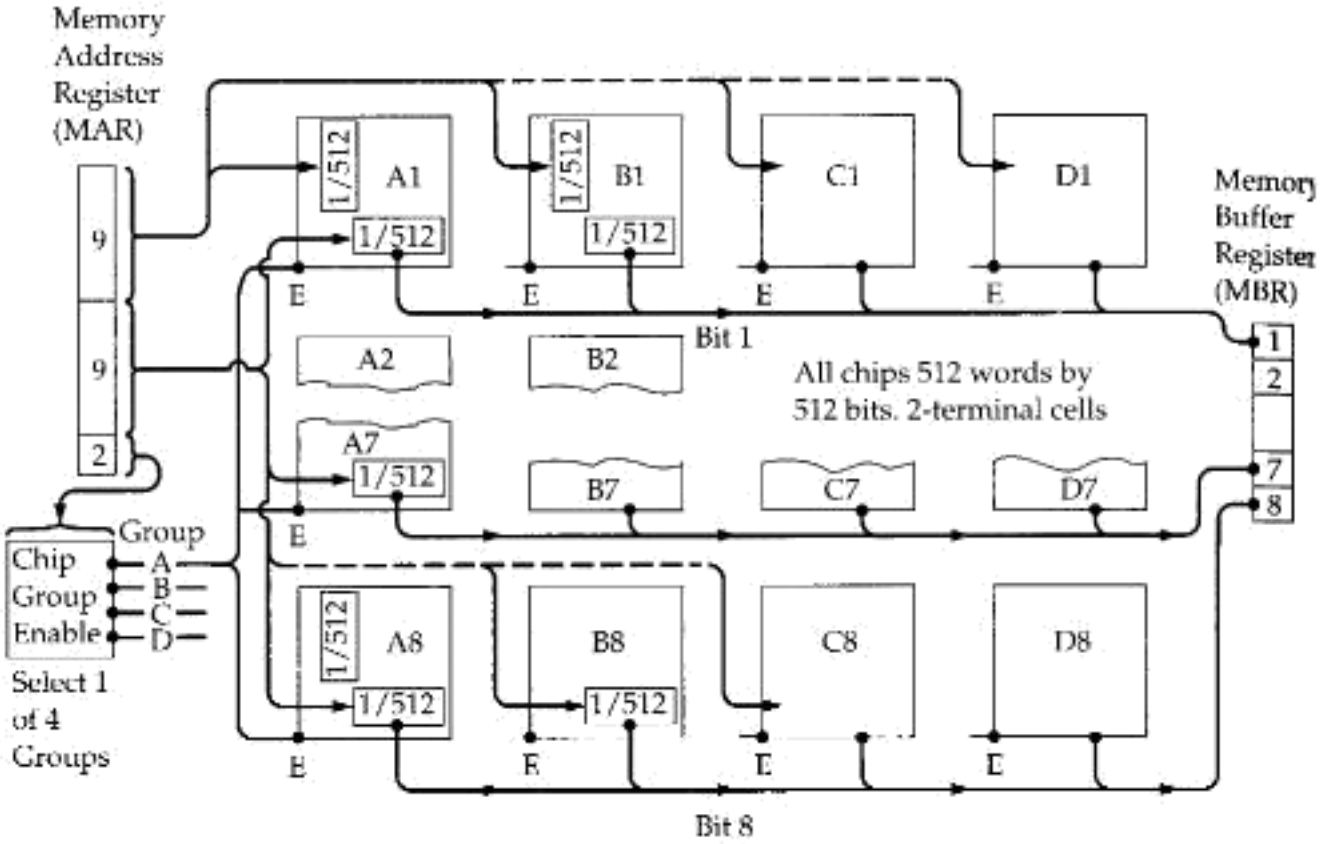


(a) 8 Mbit EPROM (b) 16 Mbit DRAM

Modül Organizasyonu(1)



Modül Organizasyonu (2)

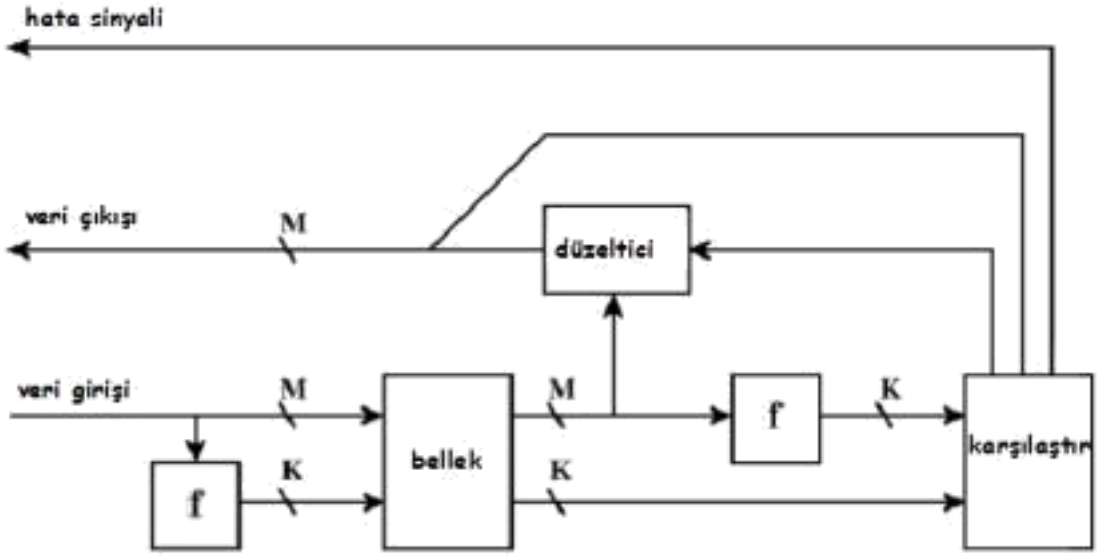


3.2 HATA DÜZELTME

Hata düzeltme (Error Correction)

- Donanım hatası (Hard Failure)
  - Kalıcı hasar verir
  - Kötü çevre koşulları
  - İmalat hatası
  - Yıpranma
- Yazılım hatası (Soft Error)
  - Rastgele her yerde olabilir, bozucu değildir
  - Belleğe kalıcı hasar vermez
  - Güç kaynağı sorunları
  - Her malzemede bulunabilen radyoaktif alfa parçacıkları
- Hamming hata düzeltme kodu ile tespit edilir

Hata düzeltme kodunun çalışması



Hamming Hata Bulma Kodu:

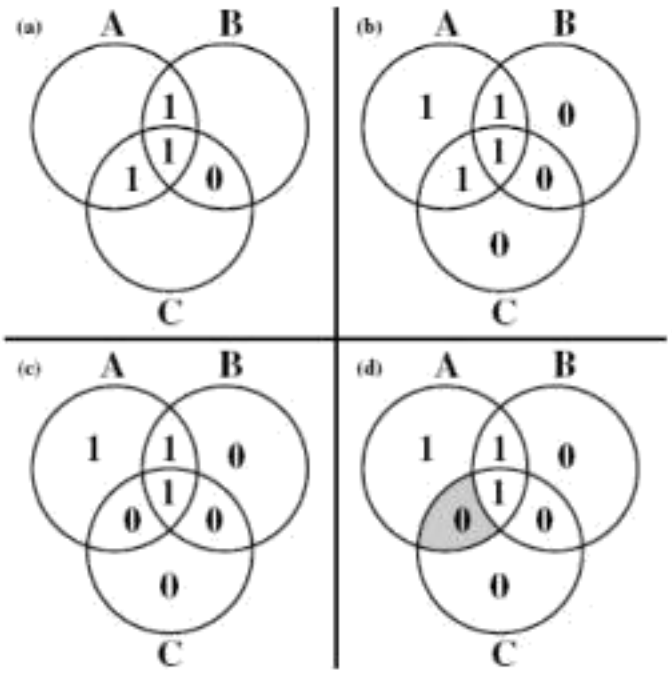


Figure 5.8 Hamming Error-Correcting Code

Bit Position	12	11	10	9	8	7	6	5	4	3	2	1
Position Number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Data Bit	D8	D7	D6	D5		D4	D3	D2		D1		
Check Bit					C8				C4		C2	C1

Figure 5.9 Layout of Data Bits and Check Bits

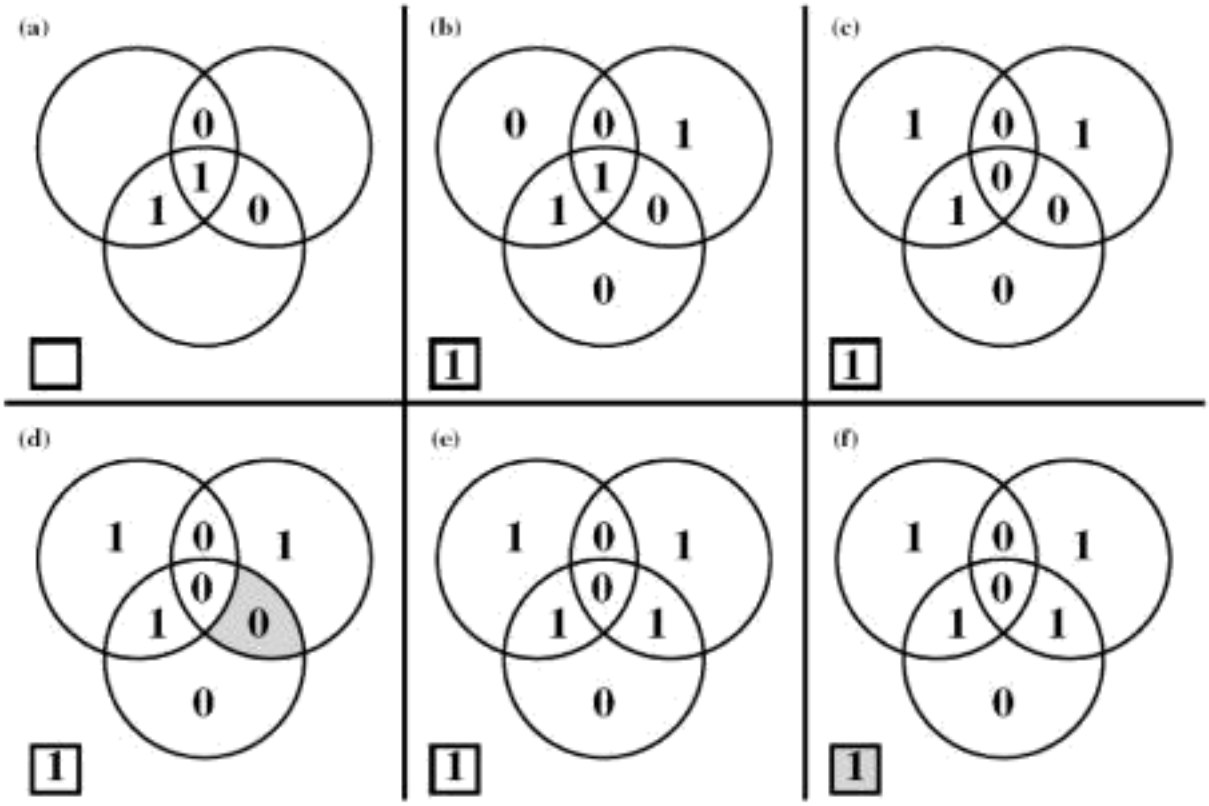


Figure 5.11 Hamming Error-Correcting Code

3.3 DAHA YENİ DRAM ORGANİZASYONU

Daha yeni RAM Teknolojileri (1)

- DRAM çipleri ilk RAM çiplerinden beri aynıdır. (1970'lerin ilk yıllarından beri. Hem iç yapısı hem de yola bağlantıları sebebiyle sınırlamaları vardır)

- İyileştirilmiş DRAM
  - İçinde az miktarda SRAM de bulunur
  - SRAM okunan son satırı tutar

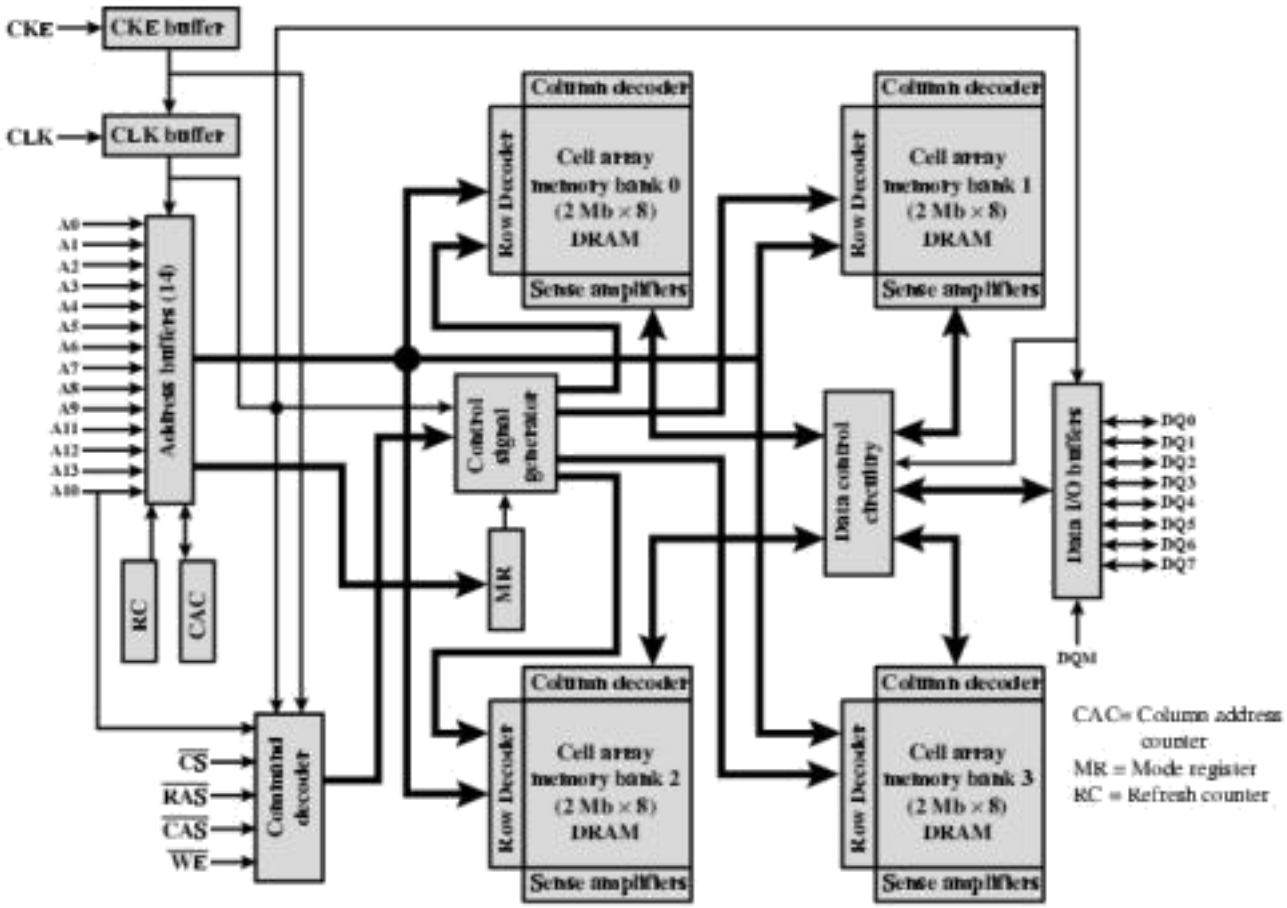
- Cache DRAM –ön-bellek DRAM--
  - Daha geniş SRAM bileşeni
  - Ön-bellek veya seri buffer olarak kullanılır

Daha yeni RAM Teknolojileri(2)

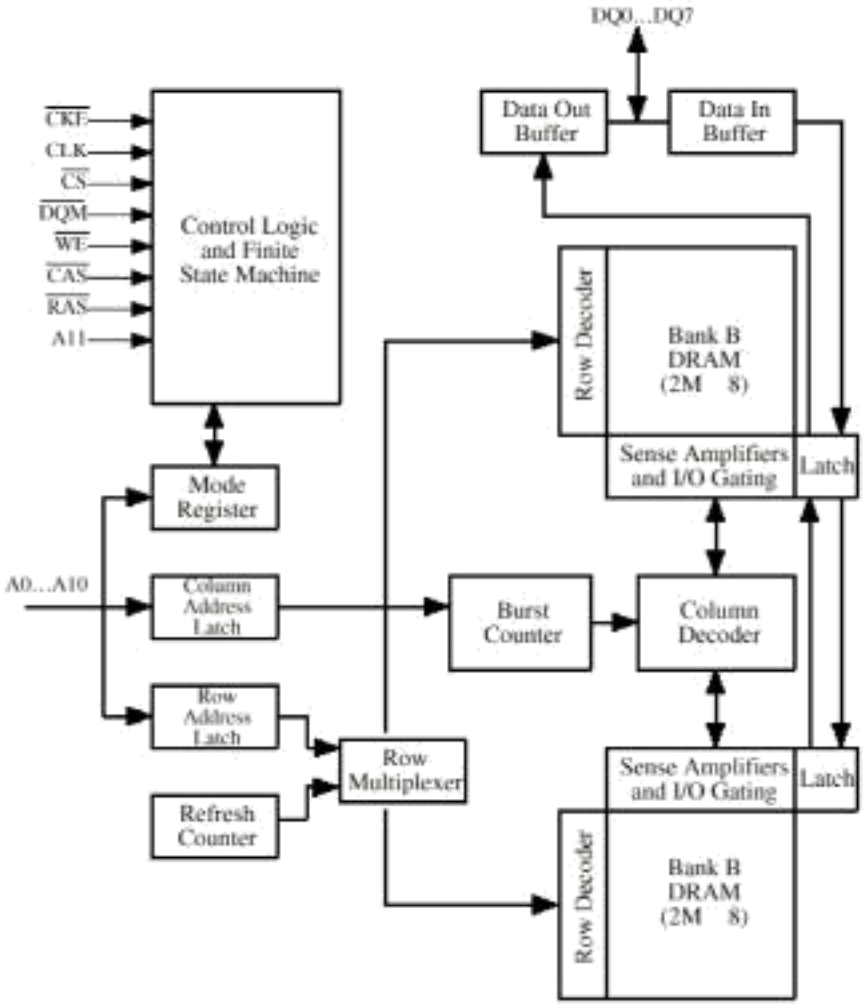
- Senkron DRAM (SDRAM)

  - DIMM 'ler üzerinde mevcuttur
  - Erişim harici bir saat ile senkronize edilir
  - Adres RAM' e gönderilir
  - RAM veriyi bulur (geleneksel DRAM'da CPU bekler)
  - SDRAM veriyi sistem saati ile senkronize olarak ilettiği için, CPU verinin ne zaman hazır olacağını bilir
  - CPU beklemek zorunda değildir, başka bir şey yapabilir
  - Burst modu sayesinde SDRAM veri dizisini blok halinde hazırlar ve dışarı gönderir
    Burst büyüklüğü 1,2,4,8 byte veya tam sayfa olabilir.

IBM 64Mb SDRAM



SDRAM



SDRAM'in Çalışması

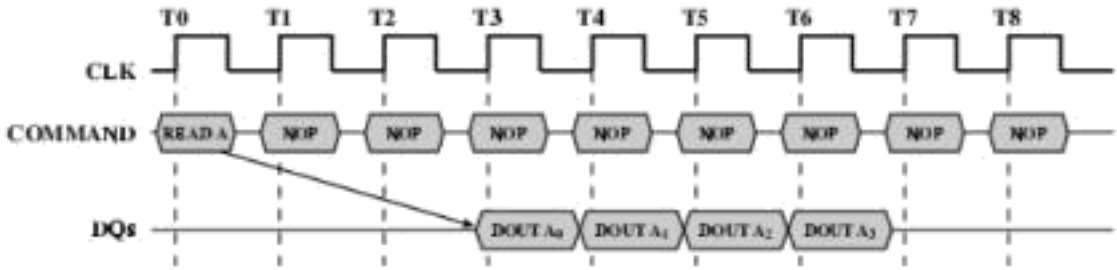
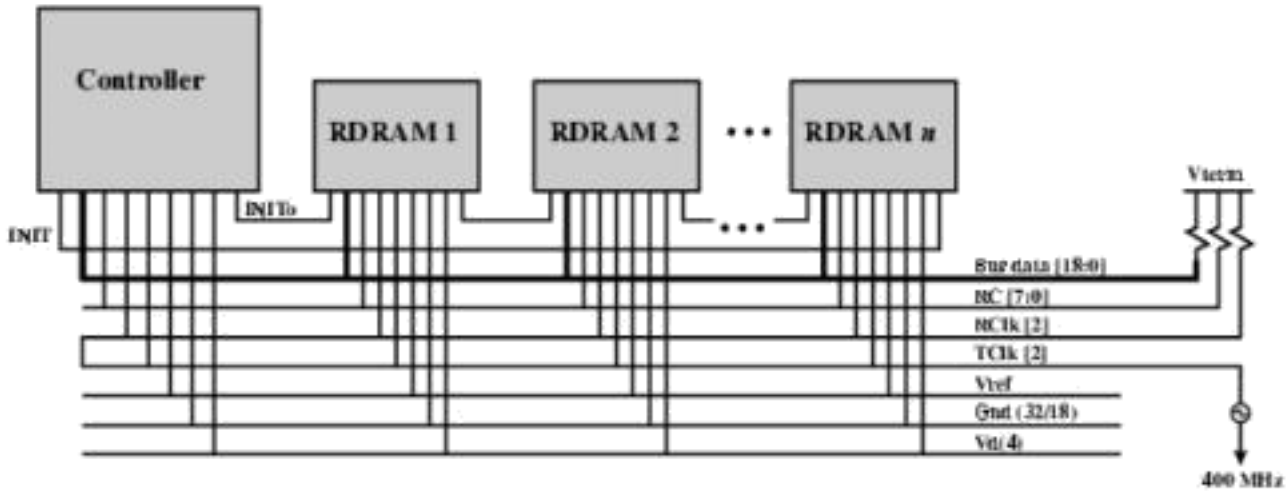


Figure 5.13 SDRAM Read Timing (Burst Length = 4, $\overline{\text{CAS}}$ latency = 2)

RAMBUS

- Pentium ve Itanium için Intel tarafından düzenlenmiştir
- SDRAM'in en büyük rakibidir
- Dikey ambalajlıdır – bütün bacakları aynı taraflıdır
- 28 hat üzerinden veri alışverişi yapar-en fazla 12cm uzunluğundadır
- Yol, 320 kadar RDRAM çipini 1.6GBps hızı ile adresler
- Asenkron blok protokolü
  - 480ns erişim süresi
  - Sonra 1.6 GBps

RAMBUS Diagramı



4. BÖLÜM

Harici Bellek (External Memory)

ÖZET:

1. Manyetik diskler halen harici belleklerin en önemli bileşeni olmaya devam ediyorlar. Taşınabilir ya da sabit, hard diskler, kişisel bilgisayarlardan süper bilgisayarlara kadar bütün sistemlerde kullanılmaktadır.

2. Daha fazla performans ve daha yüksek güvenilirlik için server ve daha büyük sistemlerde RAID disk teknolojisi kullanılır. RAID çok sayıda diskin birbirine paralel şekilde kullanıldığı, ayrıca disk hatası durumunda veriyi geri almak için fazladan disklerin dahil edildiği bir veri depolama teknolojisidir.

3. Optik depolama teknolojisi her türlü bilgisayar sisteminde giderek daha fazla önem kazanmaktadır. CD ROM yıllardır çok yaygın bir kullanıma sahipken yazılabilir CD ve DVD'ler de gittikçe önemlerini arttırmaktadırlar.

Harici Bellek Türleri

1. Manyetik Disk
   - RAID teknolojisi
   - Taşınabilir veya sabit
2. Optik
   - CD-ROM
   - Yazılabilir CD (CD-Recordable (CD-R))
   - CD-R/W
   - DVD
3. Manyetik Bant

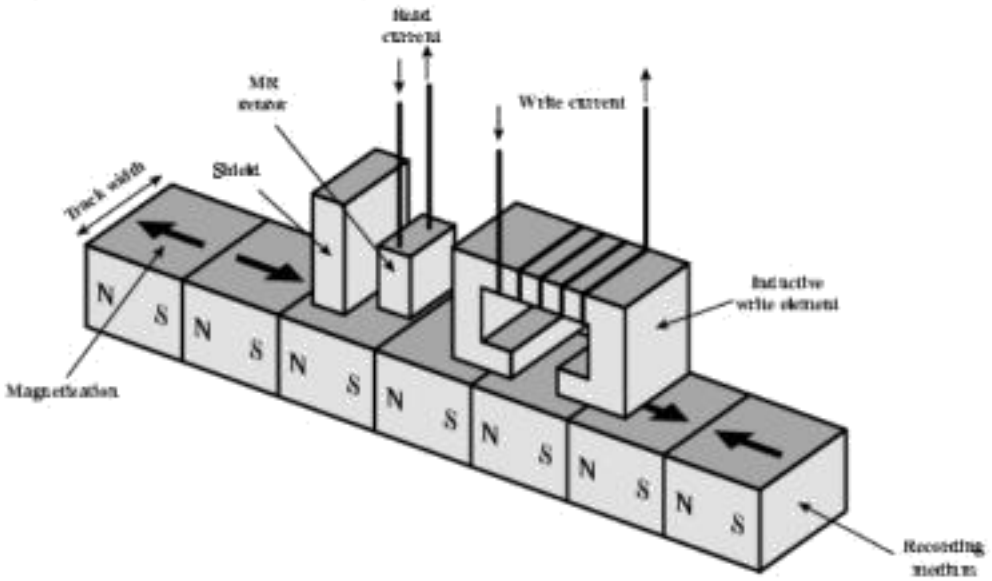
4.1 MANYETİK DİSK

- Disk mıknatıslanabilir bir malzeme ile kaplanır
- Diskin kendisi, alüminyum veya alüminyum alaşımı ile yapılırdı
- Şimdi cam kullanılır. Böylelikle:
  - Yüzey düzgünlüğü iyileşti
    - Bu da güvenilirliği artırdı
  - Yüzey hataları azaldı
    - Bu da okuma/yazma hatalarını azalttı
  - Daha düşük uçuş yüksekliği (Sonra görülecek)
  - Daha sert yapı
  - Darbelere karşı daha fazla dayanıklılık sağlandı

Oku ve Yaz Mekanizmaları

- Yazma ve okuma 'kafa' olarak adlandırılan iletken bobin üzerinden gerçekleştirilir
- Tek bir oku/yaz kafası veya ayrı ayrı kafalar olabilir
- Okuma/yazma sırasında, kafa sabit durur, disk döner
- Yazma:
  - Bobinden geçen akım manyetik alan oluşturur
  - Kafaya darbeler gönderilir
  - Manyetik bilgi kafanın altındaki yüzeye kaydedilir
- Okuma (geleneksel):
  - Manyetik alana yaklaşan bobin üzerinden alanın yönüne bağlı bir akım geçer
  - Okuma ve yazma için aynı bobin kullanılır
- Okuma (modern):
  - Yazma kafasına yakın ayrı okuma kafası
  - Kısmen koruyuculu manyeto direnç (MR) sensör
  - Sensörün direnci manyetik alanın yönüne bağlıdır
  - Yüksek frekanslı çalışma sağlar
    - Daha yüksek depolama kapasitesi ve hız

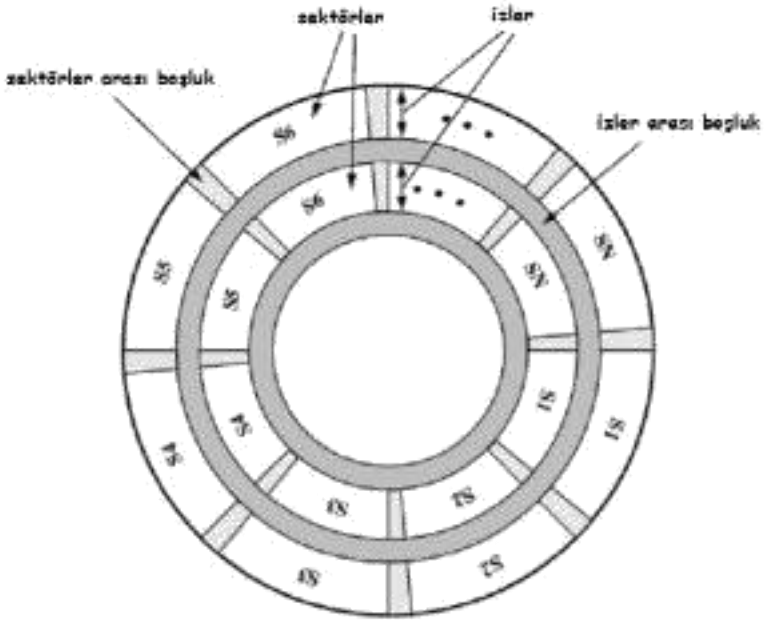
Endüktif Yazma - MR Okuma



Verinin Yerleşimi ve Biçimlendirme

- Eş merkezli halkalar veya izler (tracks)-(binlerce)
  - İzler arasında boşluklar bulunur
  - Boşlukların azaltılması kapasiteyi arttırır
  - Her iz üzerinde aynı sayıda bit (değişken yoğunluk)
  - Sabit açısal hız
- İzler sektörlere bölünmüştür (her izde yüzlerce)
- Her sektörde (büyük çoğunlukla) 512 byte bulunur
- Veri bloklar halinde (sektörler) transfer edilir

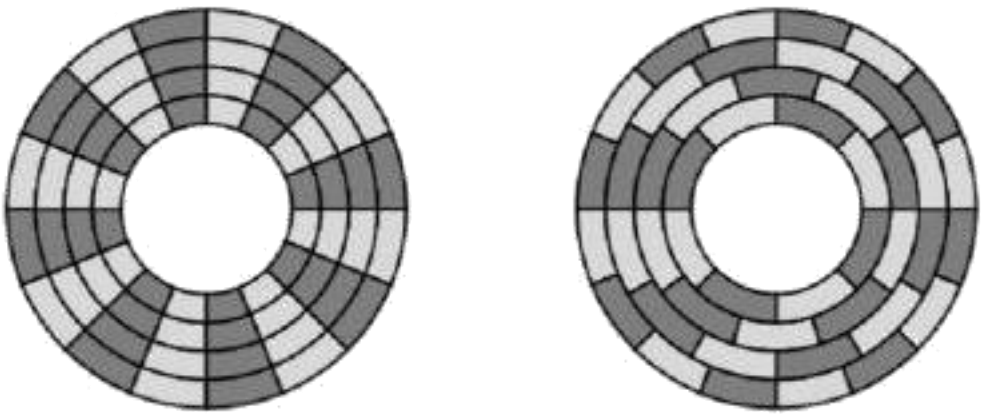
Diskte Veri Dizilimi



Disk Çalışma Hızı

- Dönen diskin merkezine yakın olan bitler, sabit bir noktadan, dış tarafta olanlara göre daha yavaş geçerler
- Farklı izlerdeki bitler arasındaki boşluklar farklı olur
- Disk sabit açısal hızda döndürülür
  - Her bir iz ve sektör adreslenebilir
  - Kafa belirlenen ize gider ve verilen sektörün gelmesini bekler
  - Daha dış izlerde boşluklar olur
    - Bu da veri yoğunluğunu düşürür
- Kapasiteyi arttırmak için bölgeleri kullanır (sabit doğrusal hız)
  - Her bölgede iz başına düşen bit sayısı sabittir
  - Daha karmaşık devreler

Disk Veri Dizilim Yöntemleri

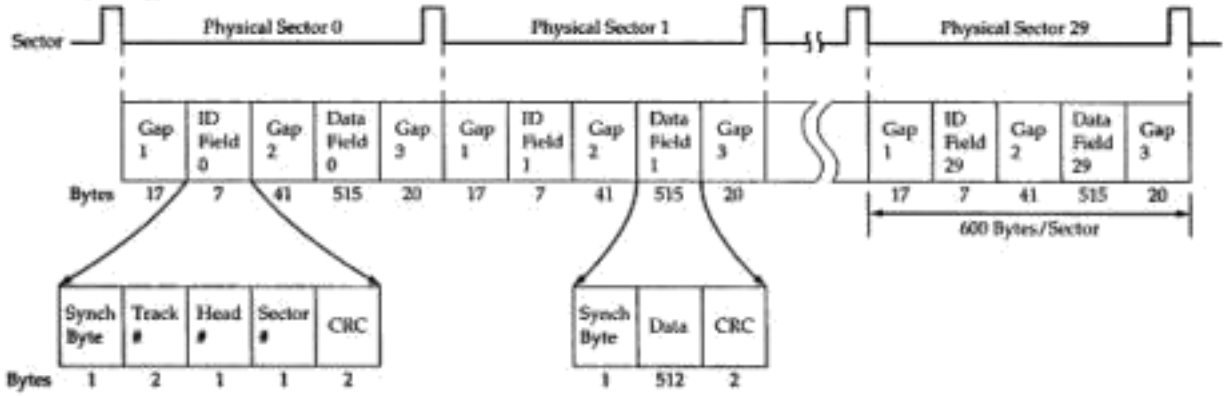


(a) Sabit açısal hız (b) Çoklu bölge kaydı

Sektörlerin Bulunması

- İz ve sektörlerin başlangıcını belirleyebilmelidir
- Diski biçimlendirme
  - Daha fazla bilgi kullanıcıya açık değildir
  - İz ve sektörleri işaretler

(Örnek)Seagate ST506 iz formatı



Fiziksel Özellikler

1. Sabit (nadir) veya hareketli kafa
2. Çıkartılabilir veya sabit olabilir
3. Tek yada (genellikle) çift yüzlü
4. Tek yada daha fazla yüzey
5. Kafa düzeneği
   - Temas (Floppy) ile okuma/yazma
   - Sabit boşluk bırakarak okuma/yazma
   - Kısa süreli (Winchester)

1. Sabit/Hareketli Kafa

- Sabit kafa
  - Her iz için bir okuma yazma kafası
  - Kafalar sabit bir kola yerleştirilmiştir
- Hareketli Kafa
  - Her yüz için bir okuma yazma kafası
  - Kafalar hareketli bir kola yerleştirilmiştir

2. Taşınabilir veya değil

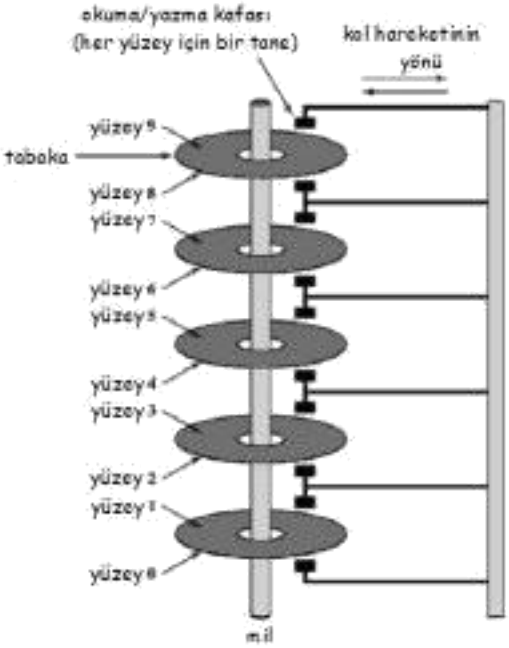
- Taşınabilir disk
  - Sürücüsünden ayrılabilir ve başka bir disk ile değiştirilebilir
  - Sınırsız depolama kapasitesi
  - Sistemler arasında kolay veri transferi sağlar
- Taşınmaz disk
  - Sürücüye sabit olarak monte edilmiştir

4. Birden fazla yüzey

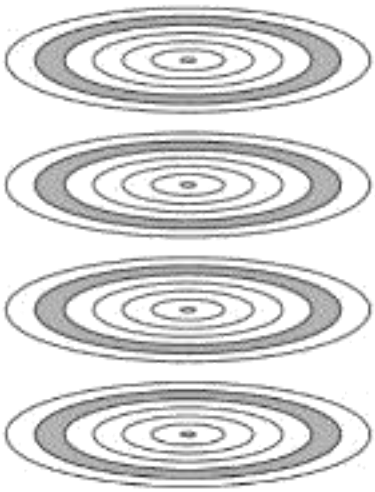
- Her yüzey için bir kafa
- Kafalar aynı kolda birleştirilmiş ve hizalanmıştır

- Her yüzdeki izler, bütün yüzeyler düşünüldüğünde bir silindir şekli oluşturur
- Veri silindir üzerindeki çizgiler şeklinde düşünülebilir. Böylelikle:
  - Kafa hareketi azalır
  - Veri transfer hızı artar

Şekil: Birden fazla yüzey



Silindirler



Floppy Disk (disket)

- 8", 5.25", 3.5"
- Düşük kapasite
  - 1.44Mbyte 'a kadar (2.88M hiç poptiler olmadı)
- Yavaş
- Çok yaygın
- Ucuz
- Eski ?

Winchester Hard Disk (1)

- Winchester (USA)'da IBM tarafından geliştirildi
- Bir ya da daha fazla yüzey (disk)
- Disk hareketsizken kafalar yüzey üzerindedirler
- Disk dönerken kafalar diske çok yakın uçarlar (diskin döntüştüntn havalandırabileceği kadar hafif metal yaprak olarak düştüntlebilirler)
- Disk boşluğuna sığması için kafalar çok küçüktür
- Bu yapı daha sağlamdır, kapasite fazladır

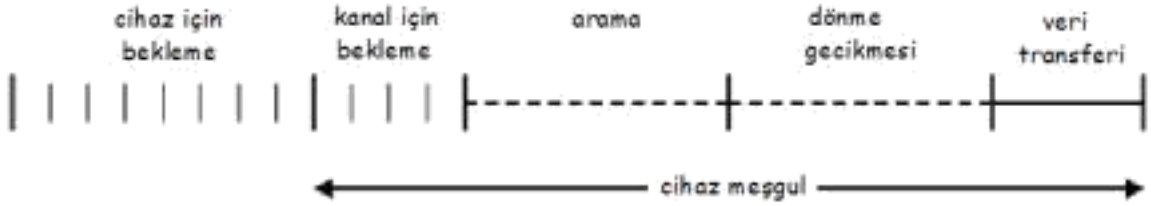
Winchester Hard Disk (2)

- Çok yaygın
- Ucuz
- En hızlı harici bellek
- Sürekli kapasitesi artıyor
  - — Gigabyte'lar çok normal

Hız

- Arama süresi (10ms'nin altındadır)
  - — Doğru izi bulmak için kafanın hareketi gerekiyor
  - — Arama süresi, kafanın doğru iz üzerine gelme süresidir
- Dönme gecikmesi(ortalama 2ms, disketler için ise 50-100ms)
- Doğru iz tizerine geldikten sonra kafanın altındaki kısmın dönmesi ve istenen sektörün gelmesi beklenir (veriyi bulmak için)
- Erişim süresi = arama süresi+ dönme gecikmesi
- Bundan sonra okuma yada yazma işlemi yapılabilir. Bu süre "Transfer süresi"dir

Disk I/O Transfer Zamanlaması



4.2 RAID Teknolojisi

RAID

- Redundant Array of Independent Disks
- Redundant Array of Inexpensive Disks
- RAID teknolojisi ile :
- -büyük kapasiteli diskler yerine, fazla sayıda daha küçük kapasiteli diskler
- -veri daha fazla sayıda diske dağıtılır ve bunlara aynı anda erişilebilir
- -böylelikle giriş/çıkış performansı artar
- -kapasite artışı kolaylaşır

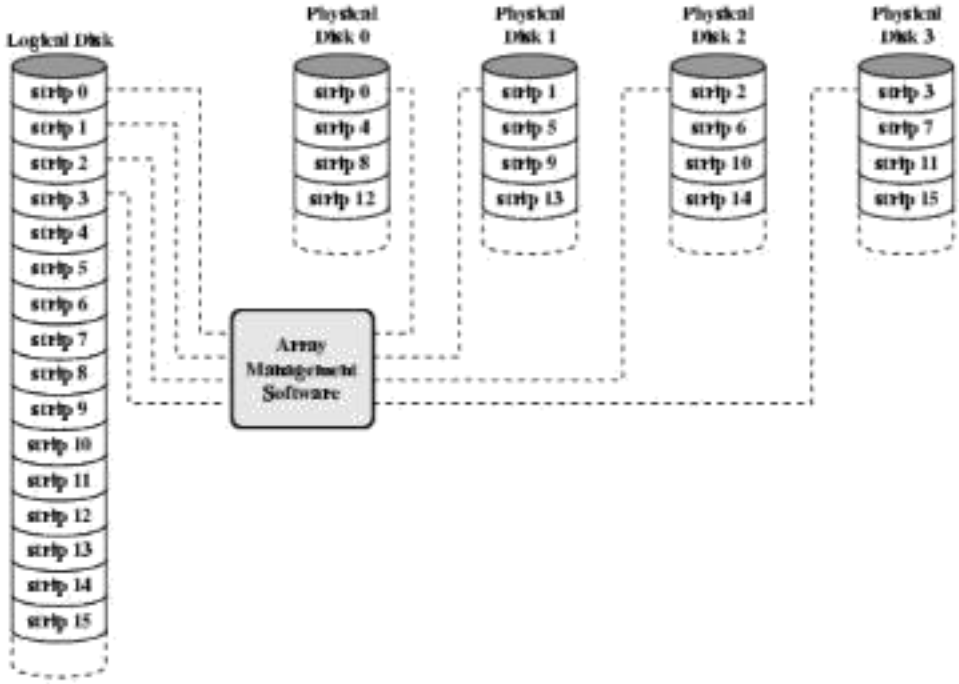
RAID

6 seviye yaygın kullanımda
Bu seviyeler hiyerarşik bir bağıntıyı göstermezler fakat farklı tasarım yapılarını belirtirler

Özellikleri:

1. RAID işletim sistemi tarafından tek bir sürücü olarak görülen bir dizi fiziksel disk sürücüsüdür
2. Veri bu fiziksel sürücüler arasında dağıtılmıştır
3. Fazla disk kapasitesi parite bilgisini saklamak için kullanılır. Böylelikle bir disk hatası durumunda veri geri alınabilir

RAID 0 için veri haritası



RAID 0

- 3.numaralı özelliği (güvenlik artırımı) desteklemez yani fazla disk kapasitesi yoktur
- Bu yüzden performansı artırma kapasitesi yoktur ve RAID ailesinden sayılmaz
- Yine de güvenilirliğin önde tutulmadığı bazı sistemlerde görülebilir
- Veri bütün diskler arasında paylaştırılmış
- Şeritler halinde saklama
- Hızda artış sağlar
    - G/Ç birimi birden fazla veri isteğinde bulunduğunda bu veriler büyük olasılıkla aynı disk üzerinde değildir
    - Diskler paralel olarak aynı anda aranır
    - Bir dizi verinin birden fazla disk üzerine dağıtılmış olma olasılığı çok yüksektir

Yüksek performans ve kapasitenin yeterli olduğu sistemlerde istenebilir.

RAID 1

- Kopyalanmış diskler
- Veri diskler üzerine dağıtılmıştır
- Her şerit grubunun farklı diskler üzerinde 2 kopyası bulunur
- Herhangi birinden okuyabilir

- Her ikisine de yazar
- Hata durumunda verinin tekrar kazanılması kolaydır
  - — Hatalı disk bilgisi hemen doğru olan ile değiştirilir
  - — Yedek veri her an hazırda olduğundan getirme süresi yoktur
- Pahalı (tek kötü yönü)

Eğer işlemler Giriş/Çıkış ağırlıklı ise ve okuma işleminde R0'ın hızını ikiye katlayabilir. Veri okuma hızı da çok yüksek olur. Fakat yazmada fazla bir üstünlüğü yoktur.

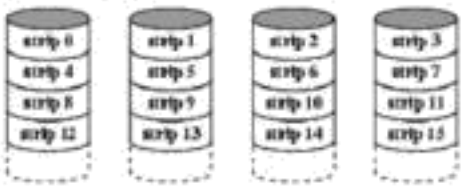
RAID 2

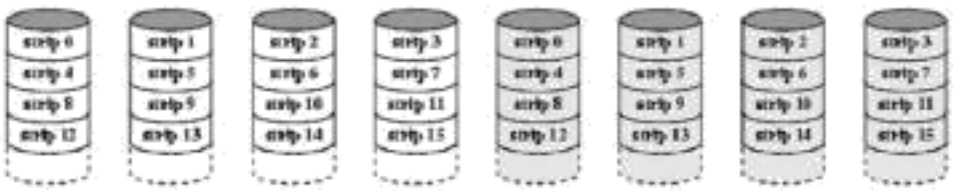
- Diskler senkronize çalışırlar (hepsindeki kafalar aynı anda aynı iz ve sektör üzerinde)
- Şeritler çok küçüktür
  - — Genellikle tek bir byte veya word
- Hata düzeltme kodu her bir disk üzerinde aynı konumdaki bitler için düzenlenir
- Hatalı bir biti bulup düzeltebilen, 2.sini ise tespit edebilen Hamming hata düzeltme kodları başka disklere yine aynı konumlara kaydedilir
- Çok sayıda fazladan disk gerektirir
  - — Pahalıdır
  - — Çok sayıda hata olma durumu karşısında etkili bir çözüm olabilirdi. Ancak günümüzün güvenilir diskleri ile gereksizdir
  - — Kullanılmaz

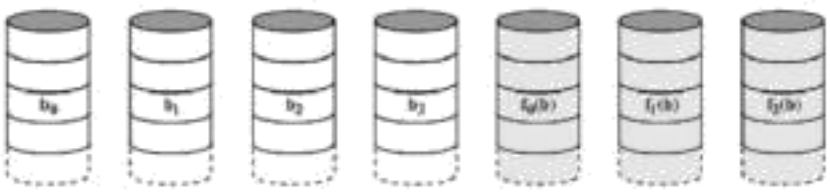
RAID 3

- RAID 2'ye benzer
- Disk sayısı ne kadar çok olursa olsun sadece bir tane fazladan disk bulunur
- Karşılıklı aynı yerlerdeki bitler için basit eşlik biti kodu kullanılır
- Disk hatası sebebiyle kaybolan veri kaybolmamış olan veri ve eşlik bilgisi sayesinde geri kazanılır
- Çok yüksek veri transfer hızları sağlar. Fakat bir anda sadece bir giriş/çıkış işlemi yapılabilir. O yüzden fazla alışveriş yapılan bir ortamda performansı iyi değildir.

RAID 0, 1, 2



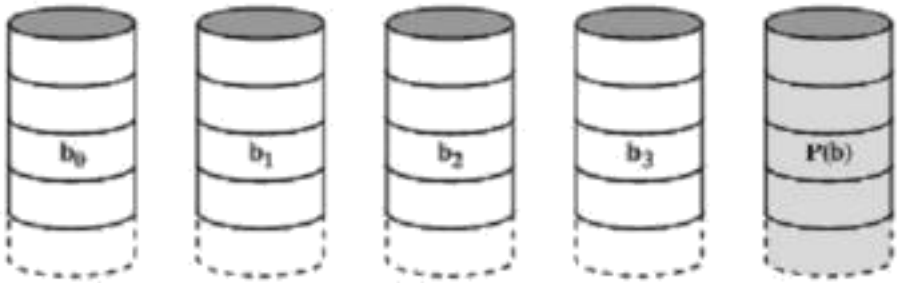




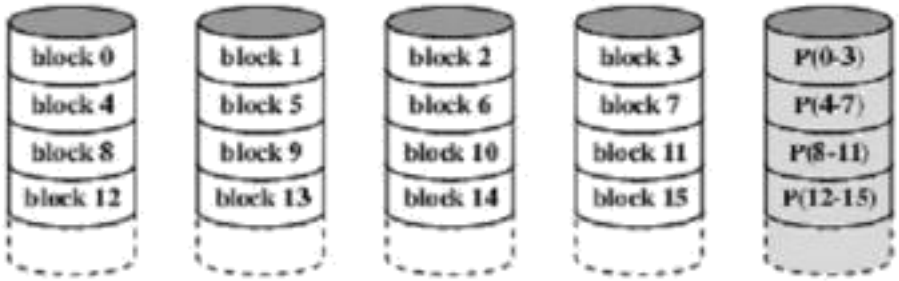
RAID 4

- Her bir disk birbirinden bağımsız çalışır
- Yüksek Giriş/çıkış isteği hızları için iyidir
- Şeritler daha geniştir
- Her diskteki şeritler üzerinde karşılıklı bitler için eşlik biti hesaplanır ve yine şerit olarak eşlik diskine kaydedilir
- Eşlik bilgisi eşlik diski üzerine diğer disklerin karşılığı olan yere kaydedilir
- Her yeni veri kaydı gerektiğinde eşlik bilgisi de gerektiği için bu durum sistemin zayıf noktası olarak görülür

RAID 3 ve 4



(d) RAID 3 (bit-interleaved parity) bit seviyesinde eşlik



(e) RAID 4 (block-level parity) blok seviyesi eşlik

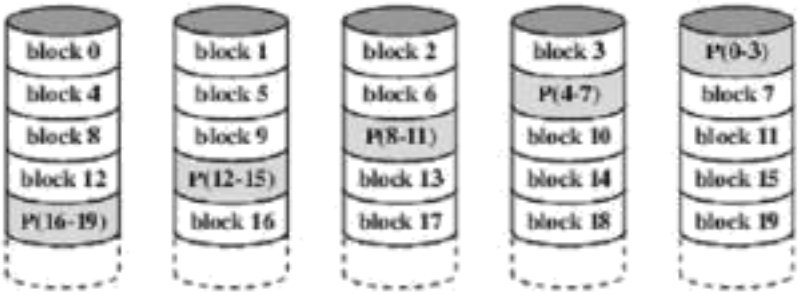
RAID 5

- RAID 4'e benzer
- Eşlik bilgisi bütün disklere dağıtılmıştır
- Eşlik şeridi de diğer şeritlerde olduğu gibi sırayı takip ederek ilgili diske kaydedilir
- RAID 4'ün kötü yönü burada giderilmiştir
- Network server'larında yaygın olarak kullanılır

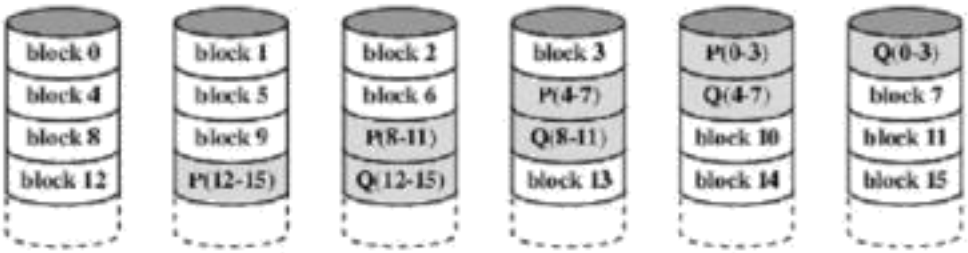
RAID 6

- İki eşlik hesabı:
- -birisi normal eşlik biti hesabı. Bu sonuç bir diske kaydedilir
- -diğerinde ise farklı bir kontrol algoritması kullanılır ve 2. fazla diske kaydedilir. Böylece birden fazla diskin kaybı telafi edilebilir
- N tane disk verisi için N+2 tane disk gereklidir
- Yüksek veri güvenliği sağlar
  - — Veri kaybı için 3 diskin hatası gerekir
  - — Yazma sırasında büyük zaman kaybı(her yazma işlemi 2 eşlik diskini de etkilediği için)

RAID 5 ve 6



(f) RAID 5 (block-level distributed parity) blok seviyesi dağıtılmış eşlik



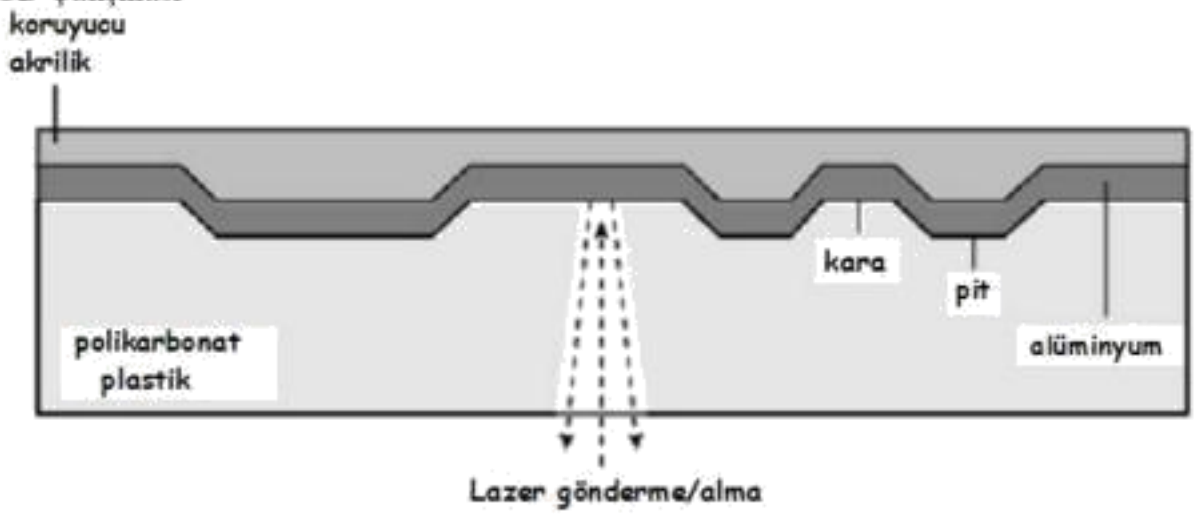
(g) RAID 6 (dual redundancy) fazladan bir çift disk

6.3 OPTİK BELLEKLER

Optik Depolama CD-ROM

- İlk olarak ses depolamak amacıyla geliştirildi(1983)
- Büyük ticari başarı kazandı
- 70 dakikadan fazla kesintisiz ses kaydedilebilen 680Mbyte kapasite
- Çok yüksek yansıtma kapasitesine sahip malzeme (genellikle alüminyum) ile kaplanmış polikarbonat
- Veri "pit"ler olarak kaydedilir
- Geri yansıyan lazer aracılığı ile okunur
- Sabit yoğunluğa sahiptir
- Sabit doğrusal hız ile çalışır

CD Çalışması



Optik Diskler

- Optik diskler, genellikle 4.75 inç (yaklaşık 12 cm) çapında ve yaklaşık 1mm. kalınlığında, lazer ışınlarıyla veri okuyan ya da yazan, taşınabilir disklerdir.

- Ayrıca, 3.5, 5.25, 8, 12 ve 14 inçlik boyutlarında da bulunur. Optik disk teknolojisinde, mekanik bir kol bulunmaz. Onun yerine, sert plastik veya metalik yüzeye, yüksek enerjili lazer ışınlarıyla küçük delikler yakılarak veri kaydedilir.

- Veriyi okumak için, düşük enerjili lazer ışınları yüzeyi tarar. Delik bulunan noktalar ışığı yansıtmaz ve bunlar ***0*** olarak yorumlanır. Düz yerler ışığı yansıtır ve bunlar ***1*** olarak yorumlanır.

- Veri optik disklerde değişik biçimlerde saklanır. Kapasiteleri 17 GB'ye kadar çıkabilir. En yaygın iki türü ;

- CD (**C**ompact **D**isk - yoğun disk)

-DVD (**D**igital **V**ersatile **D**isk veya **D**igital **V**ideo **D**isk - sayısal çok yönlü disk veya sayısal video disk)'dir.

CDler

- CD biçimi en yaygın kullanılan türdür. Çoğu kişisel bilgisayar sisteminde CD standart olarak bulunmaktadır.
- Genellikle, bir CD yüzünde 650 MB veri saklanabilir. CD'ler, CD sürücüsüne yerleştirilerek kullanılır.
- CD sürücülerin değişik dönüş hızları bulunmaktadır. Bu hız, bir sayı ve yanında X işaretiyle belirtilmektedir. "X", saniyede 150 KB veri aktarım hızını göstermektedir. Örneğin, 50X veya 50 hızlı bir CD sürücüsü, saniyede 150 KB x 50 = 7.5 MB veri aktarabilir.

Üç temel CD çeşidi vardır:

- ***CD-ROM*** (**C**ompact **D**isk-**R**ead **O**nly **M**emory - yalnız okunur bellek yoğun disk)
- ***CD-R*** (**C**ompact **D**isk-**R**ecordable - kaydedilebilir yoğun disk)
- ***CD-RW*** (**C**ompact **D**isk-**R**e**W**ritable - tekrar kaydedilebilir yoğun disk

1. CD-ROM diskler, müzik CD'leri gibidir. Disk üzerine yazılamaz veya üzerindeki veri silinemez. Yalnızca, üretici tarafından CD üzerine kaydedilen bilgiler okunabilir. Kapasiteleri 650 MB'dir. Genellikle, veritabanı veya program paketlerini dağıtmak için kullanılır

2. CD-R diskler, WORM (**W**rite **O**nce **R**ead **M**any - bir yaz çok oku) olarak da adlandırılır. CD-R disklere yalnız bir kez yazılabilir, üzerindeki veriler bozulmadan tekrar tekrar okunabilir. Bir kez yazdıktan sonra, üzerindeki veri silinemez veya değiştirilemez. CD-R sürücüler, normal müzik CD'leri ve CD-ROM'ları da okuyabilirler.

3. CD-R sürücülerin okuma ve yazma hızları değişiktir. Örneğin, 24 hızlı okuyan bir sürücünün yazma hızı 8'dir.

- CD-RW diskler, silinebilir optik diskler olarak da adlandırılır. Veri kaydedilirken disk yüzeyi kalıcı olarak değiştirilmediği için tekrar tekrar yazılabilir.
- CD-RW sürücülerin okuma, yazma ve tekrar yazma hızları değişiktir. Örneğin, 32 hızlı okuyan bir sürücünün yazma hızı 8, tekrar yazma hızı 4'dür.
- CD-RW diskleri, CD-ROM sürücüler tarafından okunamaz.
- Genellikle veri yedekleme ve çoklu ortam çalışmalarını saklamak için kullanılır

DVD'ler:

- DVD biçimi, CD biçimiyle benzer şekildedir. Ancak,veriyi belirten deliklerin boyutu CD'lerdekinden çok daha küçük ve birbirine daha yakındır. Buna ek olarak, ışığın değişik açılarda yansıması kullanılarak veri iki değişik katman halinde saklanabilir.
- Ayrıca, DVD'lerin iki yüzü de kullanılabilir. Böylece kapasiteleri, 4.7 GB'den 17 GB'ye kadar ulaşılır. Tek yüzlü ve tek katmanlı DVD 4.7 GB; tek yüzlü ve çift katmanlı DVD 8.5 GB; çift yüzlü ve tek katmanlı DVD 9.5 GB; çift yüzlü ve çift katmanlı DVD 17 GB kapasitesindedir.

DVD Çeşitleri:

- DVD-ROM (**D**igital **V**ersatile **D**isk-**R**ead **O**nly **M**emory - yalnız okunur bellek sayısal çok yönlü disk), daha çok film izlemek için kullanılır.
- DVD-R (**D**igital **V**ersatile **D**isk-**R**ecordable - kaydedilebilir sayısal çok yönlü disk), yeni, ancak pahalı bir teknolojidir. Fiyatları ucuzladıkça, çok yakın gelecekte CD-R teknolojisinin yerine geçmesi beklenmektedir.
- DVD-RW (**D**igital **V**ersatile **D**isk- **ReW**ritable - tekrar kaydedilebilir sayısal çok yönlü disk)
- DVD+RW (**D**igital **V**ersatile **D**isk+ **ReW**ritable - tekrar kaydedilebilir sayısal çok yönlü disk) gibi çeşitleri vardır. Bunlar tekrar tekrar yazılabilen ve silinebilen disklerdir.

14. BÖLÜM

Komut Seviyesi Paralellik ve Süper ölçekli (Superscalar) İşlemciler

ÖZET:

1. Süper ölçekli işlemci, birbirinden bağımsız birden fazla iş hattının kullanıldığı işlemcidir. Her iş hattında birden fazla aşama bulunur. Böylelikle her iş hattı aynı anda birden fazla komut işleyebilir. Bu özellik komut-seviyesi paralellik olarak adlandırılır ve süper ölçekli işlemciler tarafından kullanılabilir.

2. Süper ölçekli bir işlemci aynı anda birden fazla komutu alır. Sonra bu komutlara yakın olan ve bağımsız olarak işlenebilecek komutlar arar. Böylece paralel olarak çalıştırabilecektir. Eğer bir komuta gereken veri daha öncekinin sonucu olacak ise bu durumda sonraki komut diğeri ile aynı anda veya daha önce yürütülemeyecek demektir. Bu tür bağımlılıklar belirlendikten sonra işlemci komutları orijinal makine kodundan daha farklı bir sırayla işleyebilir.

3. Mikroişlemci, ek register'lar kullanarak ve orijinal kodlardaki bazı register'ların isimlerini değiştirerek gereksiz bağımlılıkları eleyebilir.

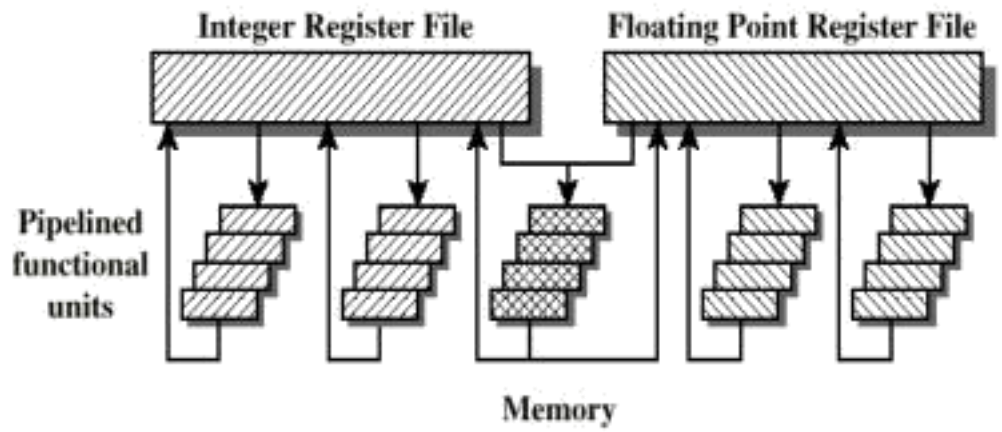
14.1 Süper ölçekli nedir?

- Aritmetik işlem komutları, load/store, koşullu atlama komutları gibi sıklıkla kullanılan komutlar bağımsız olarak alınabilir ve çalıştırılabilirler.
- Hem RISC hem de CISC mimariler ile kullanılabilir.
- Fakat uygulamada genellikle RISC ile kullanılır.

Neden Süper ölçekli ?

- İşlemlerin çoğu belirli sayısal büyüklükler iledir.
- Bu işlemlerin iyileştirilmesi bütün sistemde genel bir iyileşmeye sebep olur.

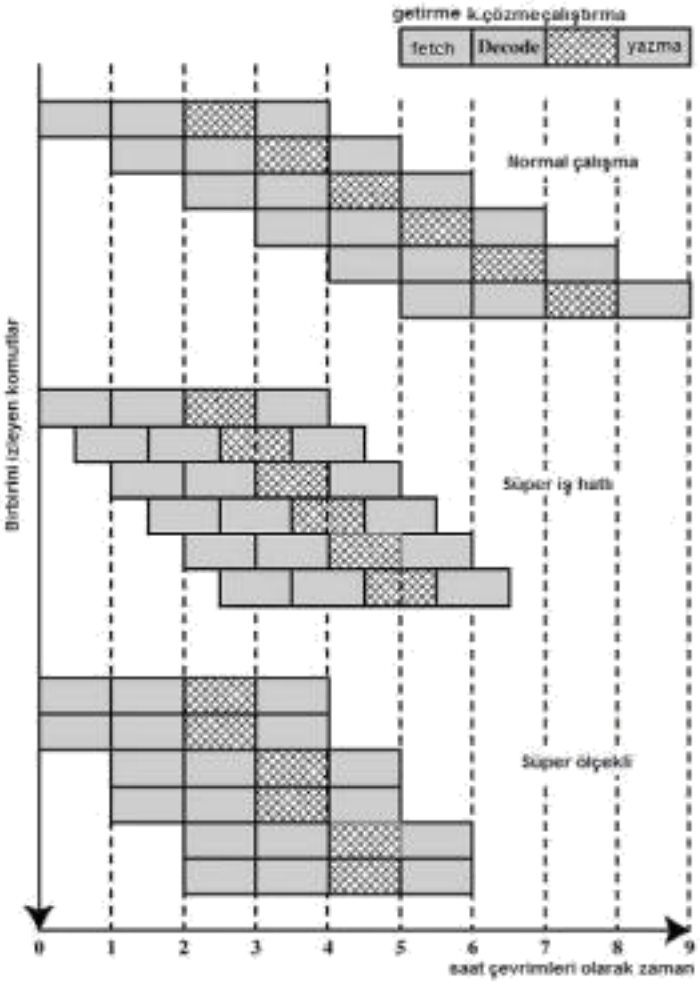
Süper ölçekli Organizasyon



Süper iş hattı

- İş hattı aşamalarının çoğu için yarım saat çevriminden daha az bir süre yeterlidir.
- Çiftlenmiş dahili saat hızı ile, her harici saat çevrimi için iki görev yerine getirilebilir.
- Süper ölçekli sistem ile komutların paralel olarak getirilmesi ve yürütülmesi mümkün olur.

Süper ölçekli mi? Süper iş hattı mı?



14.2 Süper ölçek yaklaşımının zorlukları(1)

- Süper ölçek yaklaşımı işlemcinin birden fazla komutu paralel olarak aynı anda işleyebilmesine bağlıdır.
- Komut seviyesi paralellik kavramı, bir programın komutlarının, ortalama olarak, hangi ölçüde paralel işlenebildiğini belirtir.

Süper ölçek yaklaşımının zorlukları(2)

Komut seviyesi paralelliğin performansı daha da arttırabilmesi için:

a) Derleyici tabanlı optimizasyon ve
b) Donanım teknikleri ile birleştirilir.

Süper ölçek yaklaşımının zorlukları(3)

Süper ölçekli makinelerde komut seviyesi paralelliğin arttırılabilmesi için kullanılan tasarım tekniklerinden önce sistemin baş etmesi gereken 5 paralel çalışma sorunu incelenecektir:

1. Gerçek veri bağımlılığı
2. Prosedür(işlem) bağımlılığı
3. Kaynak çakışmaları
4. Çıkış bağımlılığı
5. Bağımlılık karşıtlığı

1. Gerçek veri bağımlılığı

1.komut: ADD r1, r2 (r1 := r1+r2;)

2.komut: MOVE r3,r1 (r3 := r1;)

- İşlemci ikinci komutu birinci ile paralel olarak getirebilir ve kodunu çözebilir.
- Ancak, birinci bitene kadar ikinci komutu çalıştıramaz.

2. Prosedür(işlem) bağımlılığı

- Bir dallanma komutundan sonraki komutları önceki komutlarla paralel çalıştıramaz. Aslında dallanma komutu tamamlanmadan diğer komutlar çalıştırılamaz.

- Ayrıca, eğer komut uzunluğu sabit değilse, kaç defa gidip-getirme işleminin yapılması gerektiğinin anlaşılması için komutların kodunun çözülmesi gerekir.

- Bu da eş-zamanlı gidip-getirme işlemlerine engel olur.

- Bu konu, neden RISC veya benzeri sistemlerin süper ölçek teknolojisine daha hazır olduğunu açılıyor.(sabit uzunlukta komutlar)

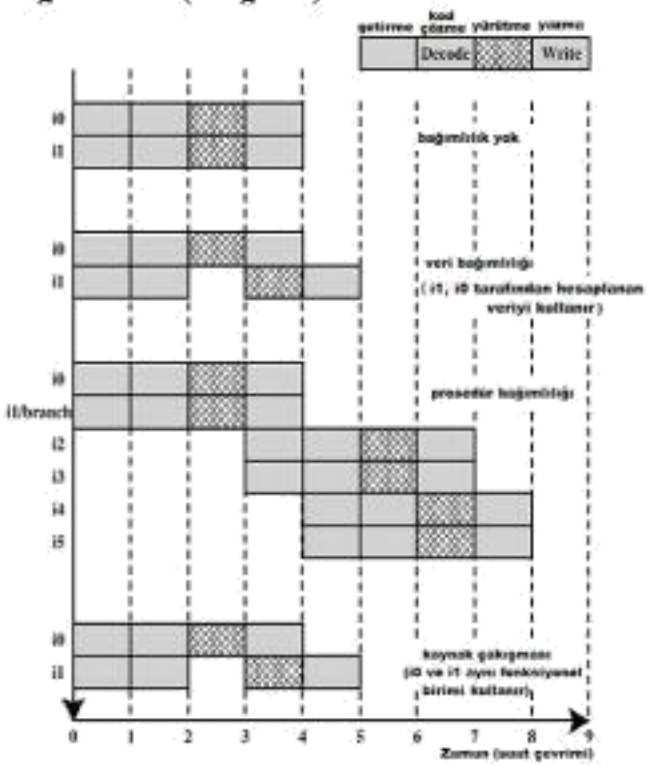
3. Kaynak çakışmaları

- İki ya da daha fazla komut aynı anda aynı kaynağa erişmek isteyebilir.(bellekler, ön bellekler, ortak yollar, portlar veya işlevsel birimler-ALU toplayıcı gibi)
  - Örneğin iki aritmetik komut

- Çözüm: Kaynakları çoğaltabilir
  - yani iki aritmetik birimi bulunabilir

(diğer iki bağımlılık konunun ilerleyen kısımlarında açıklanacaktır)

Bağımlılıklar(Diagram)



14.3 Tasarım Konuları

a) Komut Seviyesi Paralellik

Komut Seviyesi Paralellik(1)

- Birbirini izleyen bir dizi komut birbirinden bağımsızdır.
- halde bu komutlar birbirinden ayrı olarak aynı anda çalıştırılabilir.
- Veri ve işlem bağımlılığından etkilenir.

mov r1, r2	add r3, 1
add r3, 1	mov r4, (r3+r2)
add r4, r2	store [r4], r0

Soldaki üç komut birbirinden bağımsızdır ve paralel olarak çalıştırılabilirler.

Komut Seviyesi Paralellik(2)

- Sağdaki komutlar ise paralel işletilemez. Çünkü 2.komut birincinin, 3. komut ise ikincinin sonucunu veri olarak kullanacaktır.

- Komut Seviyesi Paralelliğin başarısı veri ve işlem bağımlılığının sıklığı ile ilgilidir.

- Komut Seviyesi Paralelliğin başarısı, ayrıca, bir komutun sonucunun bir sonraki komuta veri olarak yetiştirilebilmesi için geçen süreye bağlıdır.

b) Makina Paralelliği

Makina Paralelliği(1)

- Makina paralelliği, Komut Seviyesi Paralelliğin avantajını kullanabilme becerisi olarak tanımlanabilir.
- Makina paralelliği, aynı anda getirilebilen ve çalıştırılabilen komutların sayısı (paralel iş hatlarının sayısı) tarafından belirlenir.
- Makina paralelliği, işlemcinin birbirinden bağımsız komutları bulmak için kullandığı düzeneğin hızı ve gelişmişlik derecesi tarafından belirlenir.

Makina Paralelliği(2)

- Hem komut seviyesi hem de makina paralelliği performansı arttırmada önemli etkenlerdir.
- RISC sistemlerde olduğu gibi sabit uzunluktaki komutlar, komut seviyesi paralelliği kolaylaştırırlar.

14.4 Süper Ölçekli İşlemcilerde Komutların Sırası

İşlemcinin bulunduğu noktadan daha ileriye bakması ve iş hattına getirerek çalıştırabileceği komutları belirlemesi gerekir.

⇒ Komutların getirildiği sıra

⇒ Komutların çalıştırıldığı sıra

⇒ Komutların, sonuca göre, register'ları ve belleği yenilediği sıra

önemlidir.

Yöntem 1: Sırayla Al - Sırayla Çalıştır(1)

› Komutların oldukları sırayla işleme alınması.
› Çok etkili değildir.
⇒ Belekten birden fazla komut getirilebilir.
⇒ Gerektiğinde komutlar geciktirilmelidir.
⇒ Normal iş hatları bile böyle bir yöntem ile başarılı olamazlar.

Sırayla Al - Sırayla Çalıştır(2)(Diagram)

Decode		Execute			Write		Cycle
I1	I2						1
I3	I4	I1	I2				2
I3	I4	I1					3
	I4			I3	I1	I2	4
I5	I6			I4			5
	I6		I5		I3	I4	6
			I6				7
					I5	I6	8

⇒ I1'in çalışması için iki çevrim gerekiyor.
⇒ I3 ve I4 için aynı işlevsel birim gerekiyor.(kaynak çakışması)
⇒ I5'in I4 tarafından üretilen sonuca ihtiyacı var.
⇒ I5 ve I6 için aynı işlevsel birim gerekiyor.

Yöntem 2: Sırayla Al - Sırasız Çalıştır(1)(Diagram)

Decode		Execute			Write		Cycle
I1	I2						1
I3	I4	I1	I2				2
	I4	I1		I3			3
I5	I6			I4	I2		4
	I6		I5		I1	I3	5
			I6		I4		6
					I5		7
					I6		

- I2'nin I1'den önce çalıştırılmasına izin veriliyor.
- Böylelikle I3 de önce çalışabiliyor ve sonuçta program bir çevrim önce bitiriliyor.

Sırayla Al - Sırasız Çalıştır(2)

- Bu yöntemde herhangi bir anda çok sayıda komut çalıştırılabilir.
- Ancak kaynak çakışması, veri bağımlılığı veya işlem bağımlılığı durumlarında iş hatları duraklayabilir.

Sırayla Al - Sırasız Çalıştır(3)

Bu yöntemde diğer bir bağımlılık daha etkilidir: Çıkış bağımlılığı

4.Çıkış Bağımlılığı

R3← R3 + R5; (I1)
R4← R3 + 1; (I2)
R3← R5 + 1; (I3)

I2 I1'in sonucuna bağlıdır – gerçek veri bağımlılığı

Eğer I3 I1'den önce tamamlanırsa, I1'in sonucu yanlış olur - Çıkış Bağımlılığı

Bu yöntem daha karmaşık komut alma düzeneği gerektirir. Ayrıca kesmelerle ilgilenmek te daha zordur.

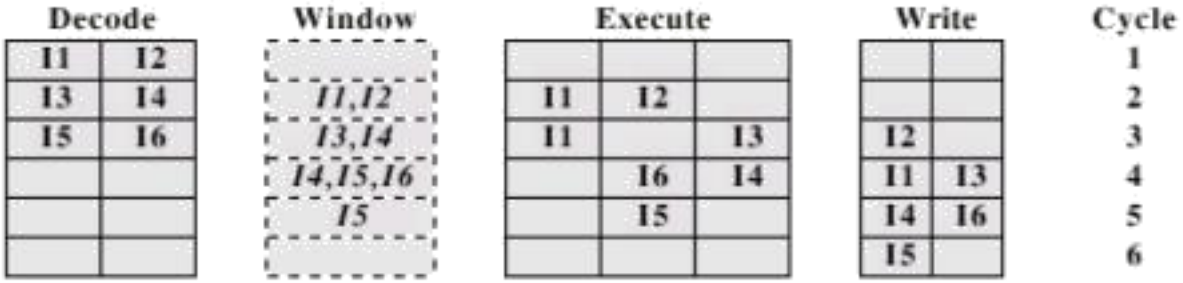
Yöntem3: Sırasız Al - Sırasız Çalıştır(1)

- Sıralı Al yöntemleriyle işlemci sadece bir sorunun oluştuğu yere kadar olan komutların kodunu çözebilir.
- Sorun çözülene kadar yeni komut kodu çözemez.
- Sonuç olarak ileri bakamaz ve zaten iş hattında olan komutlardan bağımsız olup ayrıca çalıştırılabilecek komutları getiremez.

Sırasız Al - Sırasız Çalıştır(2)

⇒ Sırasız Al yöntemi için kod çözme iş hattı ile yürütme iş hattının birbirinden ayrılması gerekir.
⇒ Böylelikle kod çözme iş hattı dolana kadar komutları getirmeye ve kodunu çözmeye devam edebilir.
⇒ İşlevsel bir birim uygun olduğunda bir komut çalıştırılabilir.
⇒ Komutların kodu çözüldüğünden işlemci ileri bakabilir.

Sırasız Al - Sırasız Çalıştır(3)(Diagram)



Burada her çevrimde iki komut getirilir ve kodu çözülür. Her çevrimde, buffer kapasitesinden dolayı, iki komut pencereye alınabilir. Bu örnekte I5 I4'ün sonucunu beklemek zorunda ama I6 değil. O yüzden I6 I5'ten önce çalıştırılabilir. Bir saat çevrimi önce bitirilmiş olur.

Sırasız Al - Sırasız Çalıştır(4)

- ⇒ Burada işlemci bir komutun kodunu çözdükten sonra onu "komut penceresi" adı verilen bir buffer'a yerleştirir.
- ⇒ Bu buffer'da yer olduğu sürece işlemci yeni komutları alıp kodlarını çözebilir.
- ⇒ İşlevsel bir birim uygun olduğunda ve herhangi bir bağımlılık sorunu olmadığında pencereden çalıştırma birimine gönderir.
- ⇒ Böylelikle işlemci "ileri bakma" özelliği kazanmış olur. Kodunu çözdüğü komutlardan bağımsız olanları ayırabilir.

Sırasız Al - Sırasız Çalıştır(5)

4. Bağımlılık Karşıtlığı (Yazma-yazma bağımlılığı)

R3←R3 + R5; (I1)

R4←R3 + 1; (I2)

R3←R5 + 1; (I3)

R7←R3 + R4; (I4)

I2'nin R3'teki veriye ihtiyacı var ve I3 R3'ü değiştireceğinden I2 çalışmaya başlayıp gerekli verileri almadan I3 işlemi bitiremez.

Bu bağımlılık gerçek veri bağımlılığına benziyor. Sadece tersi bir durum söz konusu:

ilk komutun ikinciye gerekli bir veriyi üretmesi yerine, ikinci komut birincinin kullanacağı veriyi değiştiriyor.

14.5 Register'ların tekrar isimlendirilmesi

- Sırasız Al - Sırasız Çalıştır yöntemi uygulandığında register içerikleri programdaki sıraya göre doğru olmayabileceği için "Çıkış bağımlılığı" ve "bağımlılık karşıtlığı" oluşabilir .

- İş hattının duraklamasına yol açabilir.

- Aslında bahsedilen bağımlılıklar register çakışması olarak ta görülebilir.

- Register'lar dinamik olarak tahsis edilirler.
  - yani register'lar özel olarak adlandırılmazlar

Register'ların tekrar isimlendirilmesi örneği

R3b=R3a + R5a (I1)
R4b=R3b + 1 (I2)
R3c=R5a + 1 (I3)
R7b=R3c + R4b (I4)

- Harfler düşünülmediğinde register'lar komutlarda kullanılan gerçek register'lardır.

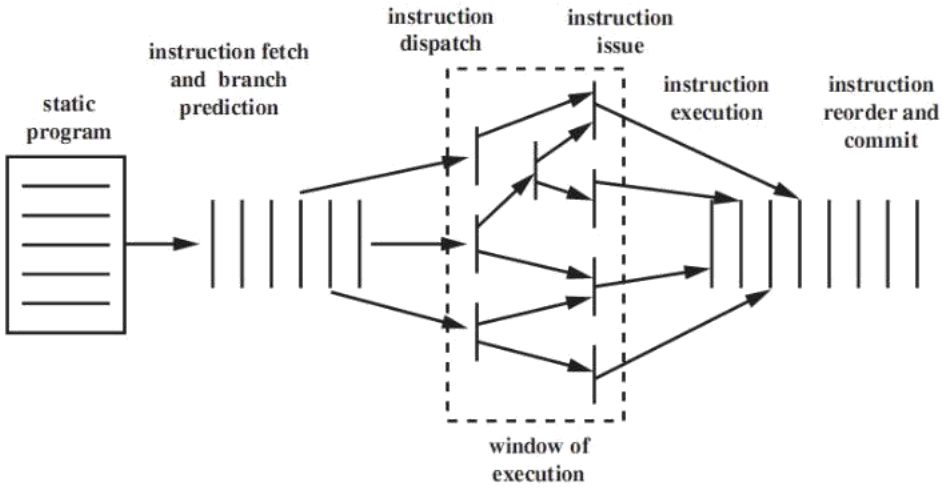
- Harflerle birlikte ise, bir register içindeki verinin her değişmesi gerektiğinde ona verilen yeni ismini gösterir.

- I3'deki R3c değişimi ile ikinci komuttaki bağımlılık karşıtlığı ve ilk komuttaki çıkış bağımlılığından kurtulunmuş olur ve I4'ün kullanacağı veri doğru belirlenmiş olur.

- Tekrar isimlendirme olmadan, ilk komut bitirilene ve ikinci komut alınana kadar I3'ün alınması mümkün olmazdı.

Bu örnekte:

I1 ile I2 arasında gerçek veri bağımlılığı var.
I1 ile I3 arasında çıkış bağımlılığı var.
I2 ile I3 arasında bağımlılık karşıtlığı var.



14.6 Makina Paralelliği (SONUÇ)

- Süper ölçekli bir işlemcide performansı arttırmak için üç farklı donanım tekniği görüldü:
    1. Kaynakların çoğaltılması
    2. Komutların Sırasız Alınması
    3. Register'ların tekrar isimlendirilmesi

- Register'ların tekrar isimlendirilmesi olmadan sadece işlevsel birimlerin çoğaltılması gerekmez. Çünkü çok az iyileştirir ama donanımı çok karmaşıklaştırır,maliyeti arttırır.(değmez yani)

- Komutların Sırasız Alınması tekniğinin daha fazla işe yaraması için komut penceresinin geniş olması iyidir. (8'den fazla)

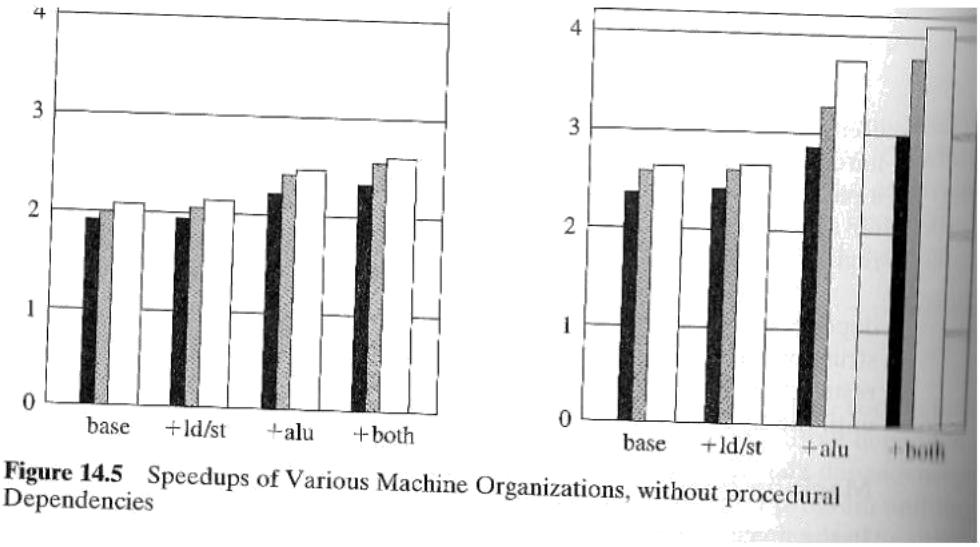


Figure 14.5 Speedups of Various Machine Organizations, without procedural Dependencies

7. BÖLÜM
Bilgisayar Aritmetiği (Computer Arithmetic)

7.1 Arithmetic & Logic Unit (ALU)

- Hesaplamaları yapar.
- Bilgisayardaki diğer her birim bu birime hizmet için çalışır.
- Tam sayılarla işlem yapabilir.
- Gerçek sayılarla işlem yapabilir.
- Bazen ayrı bir üslü sayı işlem birimi bulunabilir.
  ---FPU (maths co-processor)---
- Aynı çip üzerinde ayrı bir üslü sayı işlem birimi bulunabilir. -- FPU (486DX +)--
  (FPU-Floating Point Unit)

ALU Giriş ve Çıkışları

7.2 Tamsayıların Gösterilmesi

- Her şeyin gösterilmesi için sadece 0 ve 1'ler var.
- Pozitif sayılar ikili sistemde kolaylıkla gösterilirler.
  - Örnek : $41_{10} = 00101001_2$

- Eksi sayıları göstermek için eksi işareti yoktur.
- Virgül yoktur.

- İşaretli sayıları göstermek için iki yöntem vardır:

1. İşaret bitinin direk + veya – yi göstermesi ile
2. İkiye tümleyen yöntemi ile

1. İşaret bitinin direk + veya – yi göstermesi

- En soldaki bit işaret bitidir.
- İşaret biti 0 ise sayı pozitiftir.
- İşaret biti 1 ise sayı negatiftir.
- +18 = 00010010
- -18 = 10010010

- Bu yöntemdeki sorunlar:
  - İşlemlerde sayının hem işareti hem de değeri ayrı ayrı değerlendirilmelidir
  - 0 (sıfır) için iki farklı gösterim olur (+0 ve -0)

2. İkiye tümleyen yöntemi ile

- +3 = 00000011
- +2 = 00000010
- +1 = 00000001
- +0 = 00000000
- -1 = 11111111
- -2 = 11111110
- -3 = 11111101

İkiye tümleyen yönteminin artıları

- 0 (sıfır) için tek gösterim olur
- İşlemler kolaylıkla yapılır
- Sayıların negatifini bulmak kolaydır
  - $3_{10} = 00000011_2$
  - sayının 1'e tümleyeni 11111100
  - ikiye tümleyeni 11111101

Table 9.2 4 bitlik tam sayılar için farklı gösterim yöntemleri

Onlu sistem gösterimi	İşaretli gösterim	İkiye tümleyen gösterimi	Fazlalıklı gösterim
+8	—	—	1111
+7	0111	0111	1110
+6	0110	0110	1101
+5	0101	0101	1100
+4	0100	0100	1011
+3	0011	0011	1010
+2	0010	0010	1001
+1	0001	0001	1000
+0	0000	0000	0111
−0	1000	—	
−1	1001	1111	0110
−2	1010	1110	0101
−3	1011	1101	0100
−4	1100	1100	0011
−5	1101	1011	0010
−6	1110	1010	0001
−7	1111	1001	0000
−8	—	1000	—

Kelime uzunluklarının dikkate alınması

- Pozitif sayıların soluna sıfırlar eklenir.
- +18 – 00010010
- +18 – 00000000 00010010
- Negatif sayıların soluna birler eklenir.
- -18 – 10010010
- -18 = 11111111 10010010

Yani işaret biti aynen tekrarlanır.

7.3 Tam sayı Aritmetiği

Toplama ve Çıkarma

- Normal ikili toplama işlemi yapılır
- Taşmanın kontrolü için işaret bitine bakılır

- Çıkan sayının 2'ye tümleyeni alınarak çıkarılana eklenir
  - Yani : a - b = a + (-b)

- O halde sadece toplama ve tümleyen alma devreleri gereklidir

1001 = -7 +0101 = 5 1110 = -2 (a) (–7) + (+5)	1100 = -4 +0100 = 4 10000 = 0 (b) (–4) + (+4)
0011 = 3 +0100 = 4 0111 = 7 (c) (+3) + (+4)	1100 = -4 +1111 = -1 11011 = -5 (d) (–4) + (–1)
0101 = 5 +0100 = 4 1001 = Overflow (e) (+5) + (+4)	1001 = -7 +1010 = -6 10011 = Overflow (f) (–7) + (–6)

Figure 9.3 Addition of Numbers in Twos Complement Representation

Çarpma

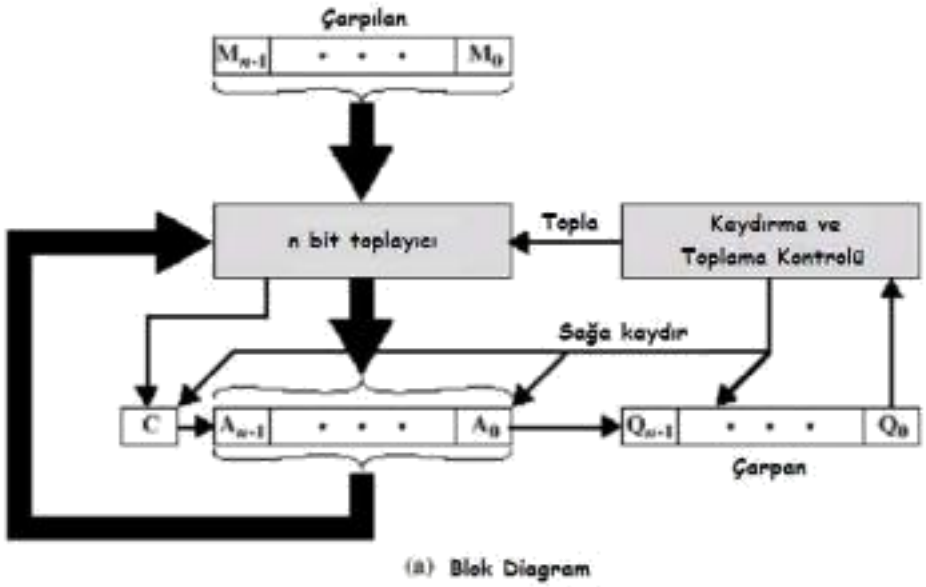
- Karmaşıktır.
- Her basamakla ayrı ayrı çarpma (kısmi çarpım) yapılması gerekir.
- Her kısmi çarpım için bir kaydırma işlemi yapılır
- Kısmi çarpımlar toplanır.

Çarpma Örneği

```
      1011   Çarpılan (onlu sistemde 11)
  x   1101   Çarpan    (onlu sistemde 13)
 ------------
      1011    Kısmi çarpım
     0000     Kısmi çarpım
    1011      Kısmi çarpım
+   1011      Kısmi çarpım
 ------------
  10001111   Çarpım (onlu sistemde 143)
```

Not: Kısmi çarpımlar, çarpan bit 1 ise çarpılanın aynısı, çarpan bit 0 ise sıfır olurlar.
Not: Sonuç için iki kat bit sayısı gerekir.

İşaretsiz Binary Çarpma



(a) Blok Diagram

Çarpma İşleminin Yürütme Örneği

C	A	Q	M		
0	0000	1101	1011	İlk değerler	
0	1011	1101	1011	Topla	Birinci çevrim
0	0101	1110	1011	Kaydır	
0	0010	1111	1011	Kaydır	İkinci çevrim
0	1101	1111	1011	Topla	Üçüncü çevrim
0	0110	1111	1011	Kaydır	
1	0001	1111	1011	Topla	Dördüncü çevrim
0	1000	1111	1011	Kaydır	

Negatif Sayıların Çarpımı

- Bir önceki yol işe yaramaz.
- Çözüm 1
  - Gerekiyorsa pozitife çevir
  - Yukarıdaki gibi çarp
  - Eğer işaretler farklıysa negatifini al
- Çözüm 2
  - Booth algoritması

Booth Algoritması Örneği

A	Q	Q_{-1}	M		
0000	0011	0	0111	İlk değerler	
1001	0011	0	0111	A = A − M	Birinci çevrim
1100	1001	1	0111	Kaydır	
1110	0100	1	0111	Kaydır	İkinci çevrim
0101	0100	1	0111	A = A + M	Üçüncü çevrim
0010	1010	0	0111	Kaydır	
0001	0101	0	0111	Kaydır	Dördüncü çevrim

```
    0111
  ×0011     (0)
11111001    1-0
0000000     1-1
000111      0-1
00010101    (21)
```
(a) (7) × (3) = (21)

```
    0111
  ×1101     (0)
11111001    1-0
0000111     0-1
111001      1-0
11101011    (-21)
```
(b) (7) × (–3) = (–21)

```
    1001
  ×0011     (0)
00000111    1-0
0000000     1-1
111001      0-1
11101011    (-21)
```
(c) (–7) × (3) = (–21)

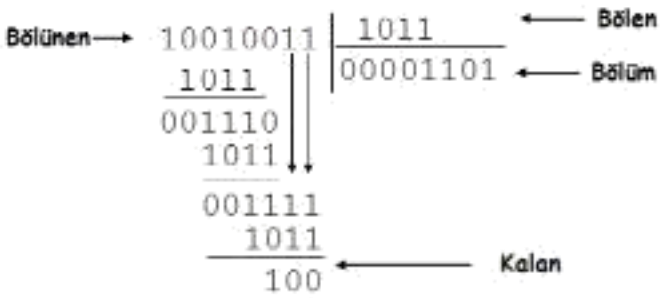
```
    1001
  ×1101     (0)
00000111    1-0
1111001     0-1
000111      1-0
00010101    (21)
```
(d) (–7) × (–3) = (21)

Figure 9.14 Examples Using Booth's Algorithm

Bölme

- Çarpmadan daha karmaşık.
- Negatif sayılarla işlem gerçekten zor.
- Uzun bölmeye dayalı.

İşaretsiz Binary Tamsayıların Bölünmesi



7.4 Üslü Sayı Gösterimi

- Tam sayıların haricinde üslü sayılara da ihtiyaç vardır.
- Çünkü tam sayılarla çok büyük ya da çok küçük sayıları yazmak mümkün olmayabilir.
- Onlu sistem sayıları ile çalışırken bilimsel gösterim kullanılarak bu sorun aşılabilir.

$$976.000.000.000.000 \quad -9{,}76 \times 10^{14}$$
$$0{,}0000000000000976 \quad -9{,}76 \times 10^{-14}$$

şeklinde yazılabilir.

- Bu yöntemle çok büyük ya da çok küçük sayılar kolaylıkla yazılabilir.

- Kesirli sayılar ikili sistemde de ifade edilebilir.

  — $1001.1010 = 2^3 + 2^0 + 2^{-1} + 2^{-3} = 9.625$

- Binary sayıdaki virgül nerede olacak?

- Nerede olduğu nasıl gösterilecek?
-

Aşağıdaki sayıların hepsi birbirine eşittir:

$0{,}110 \times 2^5$
110×2^2
$0{,}0110 \times 2^6$

Floating Point(Üslü sayılar)

işaret biti	fazlalıklı üs	Virgülden sonraki kısım (anlamlı kısım)

- Üslü sayılar "+/- .anlamlı kısım x $2^{üs}$ " şeklinde gösterilebilir.
- Virgül, işaret biti ile anlamlı kısım arasında yer alır.

Üslü Sayı Örnekleri

(a) Format

$1.1010001 \times 2^{10100}$	= 0	10010011	10100010000000000000000 = 1.6328125×2^{20}
$-1.1010001 \times 2^{10100}$	= 1	10010011	10100010000000000000000 = -1.6328125×2^{20}
$1.1010001 \times 2^{-10100}$	= 0	01101011	10100010000000000000000 = 1.6328125×2^{-20}
$-1.1010001 \times 2^{-10100}$	= 1	01101011	10100010000000000000000 = $-1.6328125 \times 2^{-20}$

(b) Örnekler

Üslü sayıların gösterimi

- En soldaki bit işaret bitidir (0–pozitif, 1–negatif)
- Üs ise sonraki 8 bite "fazlalıklı formatta" (biased notation) kodlanmıştır
  - üs için 8 bit'lik alan ayrılmıştır.
  - gerçek üs değerinin bulunması için 8 bitlik bu alandaki sayıdan "fazlalık" olarak adlandırılan sabit bir sayının çıkarılması gerekir.
  - Kayıtlı değer 0-255 aralığında olmakla birlikte gerçek değerin bulunması için 127 çıkarılır ($2^{k-1} - 1$)
  - Böylelikle gerçek değer (-128 +127) aralığında olur.
- Sonraki 23 bit sayının anlamlı kısmı yani virgülden sonraki kısmıdır.

Normalize İşlemi(1)

- Üslü sayılarla yapılacak aritmetik işlemlerin basitleştirilmesi için üslü sayıların normalize edilmeleri gerekir.
- Yani üs öyle ayarlanır ki sayının en ağırlıklı biti 1 olur
- Bu bit daima 1 olacağından yazılması gerekmez.
- Yani virgüllü sayımızın virgülden önceki tam kısmı her zaman "1"dir.
- Bu gösterim, ikili sistem sayıları için bilimsel gösterimdir.

(Onlu sistem üslü sayılarında da bilimsel gösterim, virgülden önce tek rakam olacak şekilde normalize edilmesi şeklindedir.

örneğin 3.123×10^3)

Normalize İşlemi(2)

- Normalize edilmemiş bir ikili sistem sayısının normalize edilmesi için virgül, en solda yer alan "1"in sağına alınır ve üs buna göre ayarlanır.

$$0{,}0001101 \times 2^6 = 1{,}101 \times 2^2$$
$$111001{,}0101 = 1{,}110010101 \times 2^5$$
$$0{,}0001101 \times 2^{-3} = 1{,}101 \times 2^{-7}$$
$$0{,}0001101 \times 2^2 = 1{,}101 \times 2^{-2}$$

Normalize İşlemi(3)

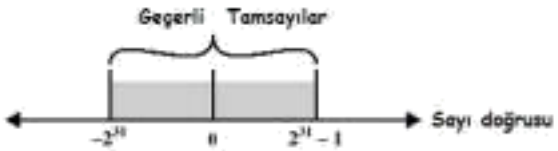
- Normalize İşleminin faydaları:

1. Veri aktarımını kolaylaştırır.
2. Aritmetik algoritmaları kolaylaştırır. Çünkü herkes sayı gösteriminin bu şekilde olduğunu bilir.
3. Sayıların hassasiyetini arttırır. Çünkü sayıların başında "sıfırlar" yoktur.

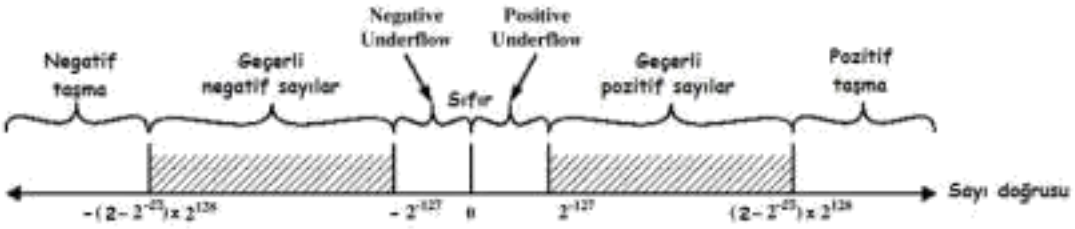
Üslü sayı aralığı(FP Ranges)(1)

- Üs için daha çok bitin kullanılması sayı aralığını arttırır.
  - 8 bit üs için
  - $+/- 2^{256} \approx 1{,}5 \times 10^{77}$
  - Yani onlu sistemde $(-1{,}5 \times 10^{77} \quad +1{,}5 \times 10^{77})$ aralığını ifade eder
- Hassasiyet
  - Anlamlı kısmın en düşük ağırlıklı bitinin değiştirilmesi ile oluşan fark
  - 23 bit anlamlı kısım için $2^{-23} \approx 1{,}2 \times 10^{-7}$
  - Yaklaşık 6 onlu sistem basamağı kadar
- Anlamlı kısım için daha çok bitin kullanılması hassasiyeti arttırır.

Geçerli Tam ve Üslü Sayılar



(a) İkiye tümleyen Tamsayılar



(b) Üslü Sayılar

Üslü sayı aralığı(2)

- 32 bitlik tam sayı aralığı:

$2^{32} - 4\,294\,967\,296 \qquad 2^{31} - 2\,147\,483\,648$

O halde

$-2\,147\,483\,648$ ile $+2\,147\,483\,647$ aralığı geçerlidir.

(8 bitlik sayılarla hepsi pozitif olan 0-255 aralığı, veya işaretli olarak -128 +127 aralığının geçerli olduğunu hatırlayalım.)

Üslü sayı aralığı(3)

- 32 bitlik üslü sayı aralığı:

Pozitif üslü sayıların en küçüğü :

$1{,}00000000000000000000000 \times 2^{-127} = 2^{-127}$

Pozitif üslü sayıların en büyüğü :

$1{,}11111111111111111111111 \times 2^{128} = (2-2^{-23}) \times 2^{128}$

Negatif üslü sayıların en büyüğü :

$-1{,}00000000000000000000000 \times 2^{-127} = -2^{-127}$

Negatif üslü sayıların en küçüğü :

$-1{,}11111111111111111111111 \times 2^{128} = -(2-2^{-23}) \times 2^{128}$

Üslü Sayıların Yoğunluğu(1)

- Üslü sayı gösterim yöntemi ile daha fazla sayının ifade edilmediğine dikkat edilmelidir. 32 bit ile hala 2^{32} farklı sayı gösterilebilir. Sadece gösterilen sayı aralıkları farklılaşmıştır.
- Ayrıca, üslü olarak ifade edilen sayıların sayı çizgisi üzerindeki aralıklarının eşit olmadığına da dikkat edilmelidir.

Üslü Sayıların Yoğunluğu(2)

- Üslü olarak gösterilen sayılar, orijine yakın kısımlarda birbirlerine daha yakındır. Orijine uzak kısımlarda ise aralıkları daha fazladır.
- Pek çok işlemin sonucu tam olarak ifade edilemediğinden en yakın sayıya yuvarlanmak durumundadır.

Üslü Sayıların Yoğunluğu(3)

- 32 bit veri ile ilgili bir başka konu da şudur:

32 bit'in 8'i üs için 23'ü ise virgülden sonraki kısım için kullanılmaktadır. Eğer üs için kullanılan bit sayısı arttırılırsa ifade edilen sayıları aralığı artar. Fakat sabit sayıda rakam ifade edilebildiği için sayı çizgisi üzerinde sayıların yoğunluğu, dolayısı ile de hassasiyeti azalır.

- Hem hassasiyeti hem de sayı aralığını arttırmanın tek yolu kullanılan bit sayısını arttırmaktır. O yüzden bilgisayarlar hem "single-32 bit" hem de "double-64 bit" veri tiplerini kullanabilecek şekilde düzenlenmişlerdir.

IEEE 754

- En önemli üslü sayı gösterim standardıdır.
- Bu standart hem programların bir bilgisayardan diğerine taşınabilmesi hem de aritmetik programların geliştirilmesini teşvik etmek için düşünülmüştür.
- Çok geniş ölçüde kabul görmüştür ve bütün yeni işlemci ve yardımcı işlemcilerde kullanılmaktadır.
- 32 ve 64 bit'lik veriler için, üsler, sırasıyla 8 ve 11 bit'litir.
- Hem üs hem de sayının kendi için ara sonuçlar için genişletilmiş format.

IEEE 754 Formatları(1)

(a) Single format

(b) Double format

- Genişletilmiş format ara sonuç değerleri için kullanılır ve tam formatı ilgili uygulamaya özel olarak belirlenir.

IEEE 754 Formatları(2)

Table 9.3 IEEE 754 Format Parameters

	Format			
Parameter	Single	Single Extended	Double	Double Extended
Word width (bits)	32	≥ 43	64	≥ 79
Exponent width (bits)	8	≥ 11	11	≥ 15
Exponent bias	127	unspecified	1023	unspecified
Maximum exponent	127	≥ 1023	1023	≥ 16383
Minimum exponent	−126	≤ −1022	−1022	≤ −16382
Number range (base 10)	10^{-38}, 10^{+38}	unspecified	10^{-308}, 10^{+308}	unspecified
Significand width (bits)*	23	≥ 31	52	≥ 63
Number of exponents	254	unspecified	2046	unspecified
Number of fractions	2^{23}	unspecified	2^{52}	unspecified
Number of values	1.98×2^{31}	unspecified	1.99×2^{63}	unspecified

Table 9.4 IEEE 754 Üslü Sayıları

	Tek hassasiyet (Single-32 bit)				Çift hassasiyet (Double-64 bit)			
	İşaret	Fazlalıklı üs	Kesir	Değer	İşaret	Fazlalıklı üs	Kesir	Değer
pozitif sıfır	0	0	0	0	0	0	0	0
negatif sıfır	1	0	0	–0	1	0	0	–0
pozitif sonsuz	0	255 (all 1s)	0	∞	0	2047 (all 1s)	0	∞
negatif sonsuz	1	255 (all 1s)	0	$-\infty$	1	2047 (all 1s)	0	$-\infty$
quiet NaN	0 or 1	255 (all 1s)	$\neq 0$	NaN	0 or 1	2047 (all 1s)	$\neq 0$	NaN
signaling NaN	0 or 1	255 (all 1s)	$\neq 0$	NaN	0 or 1	2047 (all 1s)	$\neq 0$	NaN
positive normalized nonzero	0	$0 < e < 255$	f	$2^{e-127}(1.f)$	0	$0 < e < 2047$	f	$2^{e-1023}(1.f)$
negative normalized nonzero	1	$0 < e < 255$	f	$-2^{e-127}(1.f)$	1	$0 < e < 2047$	f	$-2^{e-1023}(1.f)$
+ normalize edilmemiş	0	0	$f \neq 0$	$2^{e-126}(0.f)$	0	0	$f \neq 0$	$2^{e-1022}(0.f)$
- normalize edilmemiş	1	0	$f \neq 0$	$-2^{e-126}(0.f)$	1	0	$f \neq 0$	$-2^{e-1022}(0.f)$

7.5 Üslü sayılarla toplama/çıkarma (FP Arithmetic +/-)

- Sayıların sıfır olup olmadığı kontrol edilir.
- Üsler ayarlanarak sayıların tam kısmı eşitlenir.
- Sayıların tam kısımları toplanır veya çıkarılır.
- Sonuç normalize edilir.

Üslü sayılarla çarpma/bölme (FP Arithmetic x/÷)

- Sayıların sıfır olup olmadığı kontrol edilir.
- Üsler toplanır/çıkarılır.
- Sayılar çarpılır/bölünür (işarete dikkat edilerek).
- Normalize edilir.
- Yuvarlanır.
- Bütün ara sonuçlar "double" olarak kaydedilir.