Bölüm 1: Giriş

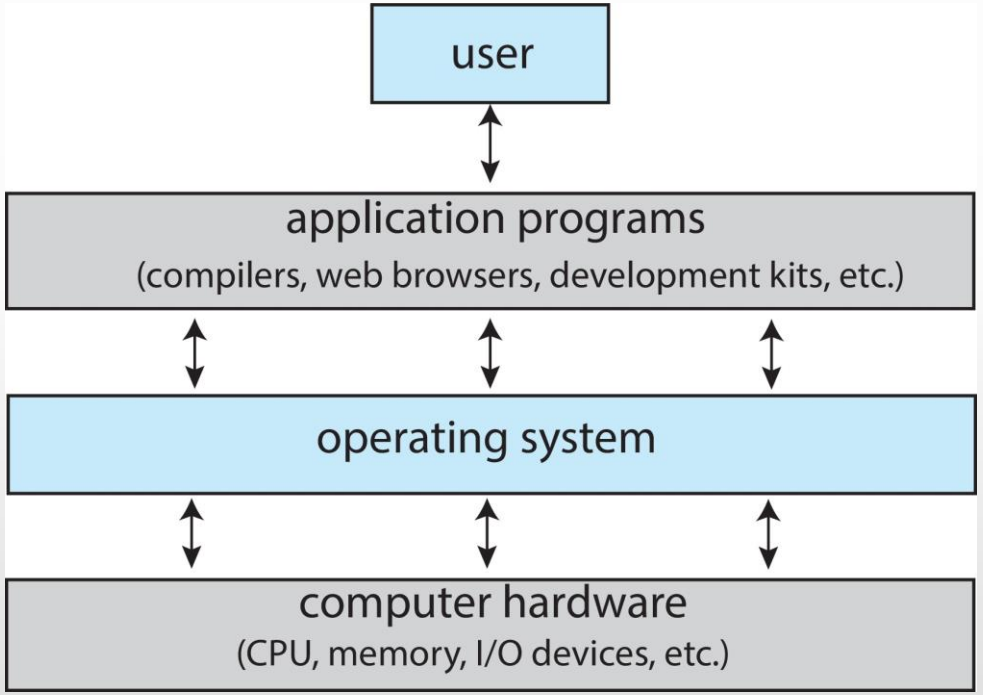
İşletim Sistemleri

İşletim Sistemi

- Modern bilgisayar çok karmaşıktır.
- Uygulama programcısının her detayı bilmesi imkansızdır.
- Kaynakları daha iyi, daha basit, ve daha sade yönetebilmek için bir katman
- Çeşitli işletim sistemleri; Windows, Linux, MacOS
- Kullanıcılar, kabuk (Shell) veya GKA (GUI) sayesinde bilgisayar ile etkileşime girer.

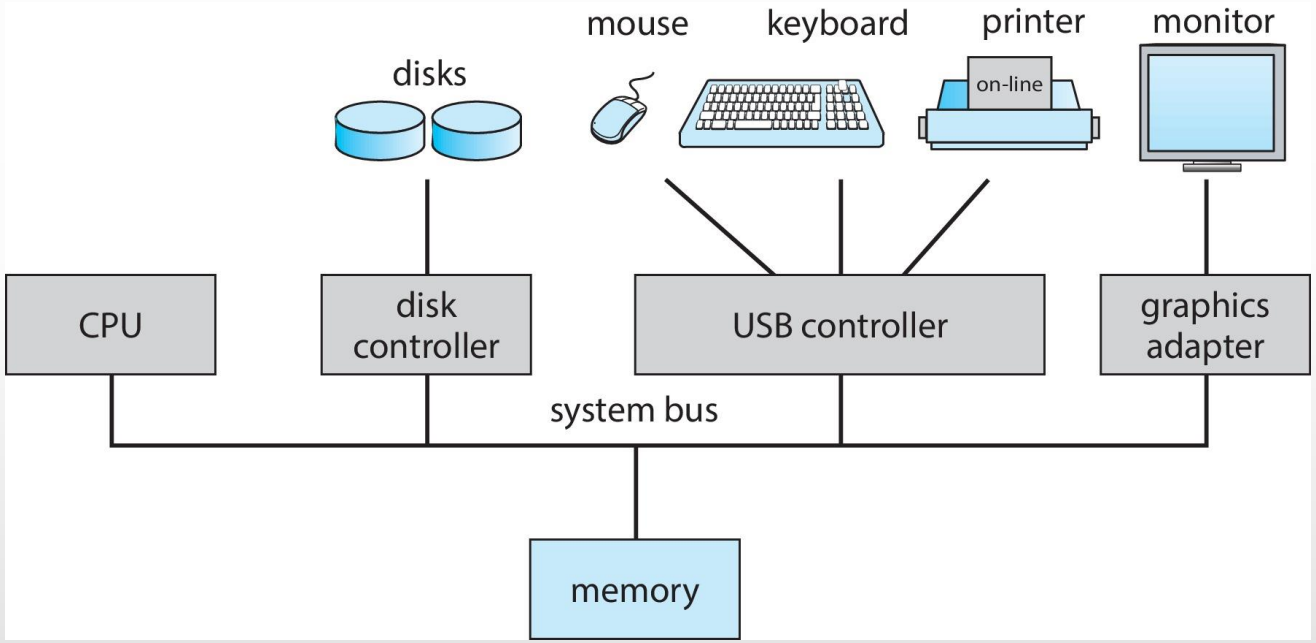
Modern Bilgisayarın Bileşenleri

- .



Modern Bilgisayarın Donanım Bileşenleri

- İşlemci
- Ana bellek
- Disk
- Yazıcı
- Klavye
- Fare
- Ekran
- Ağ arayüzleri
- G/Ç cihazları



Kabuk (Shell) ve GKA (Grafik Kullanıcı Arayüzü)

- Kabuk: Kullanıcıların işletim sistemiyle etkileşimini sağlayan komut satırı arayüzü.
- GUI (Grafik Kullanıcı Arayüzü): Simgeler, pencereler ve fare gibi grafik öğeleri kullanarak kullanıcıların işletim sistemiyle etkileşime girmesini sağlayan görsel bir arayüz.
- Kabuk ve GUI, çekirdek işletim sisteminin değil, kullanıcı arabiriminin parçasıdır.
- Kabuk ve GUI, kullanıcının işletim sistemiyle etkileşime girmesi için bir araç sağlar, ancak temel sistem işlevlerinden veya hizmetlerinden herhangi birini sağlamazlar.

Kabuk (Shell) ve GKA (Grafik Kullanıcı Arayüzü)

- Aynı işletim sisteminde aynı anda birden çok kabuk ve GUI kullanılabilir.
- Kullanıcının işletim sistemi deneyimini kişiselleştirmesine ve onu kendi özel ihtiyaçları için daha verimli ve etkili hale getirmesine olanak tanır.
- Kabuk, belirli görevleri gerçekleştirmek için daha verimli ve etkili bir yol sağlarken, GUI, işletim sistemiyle etkileşimde bulunmak için daha kullanıcı dostu ve sezgisel bir yol sağlar.
- Kabuk ve GKA, işletim sisteminin teknik yetenekleri veya tercihleri ne olursa olsun daha geniş bir kullanıcı yelpazesi tarafından erişilebilir ve kullanılabilir olmasını sağlar.

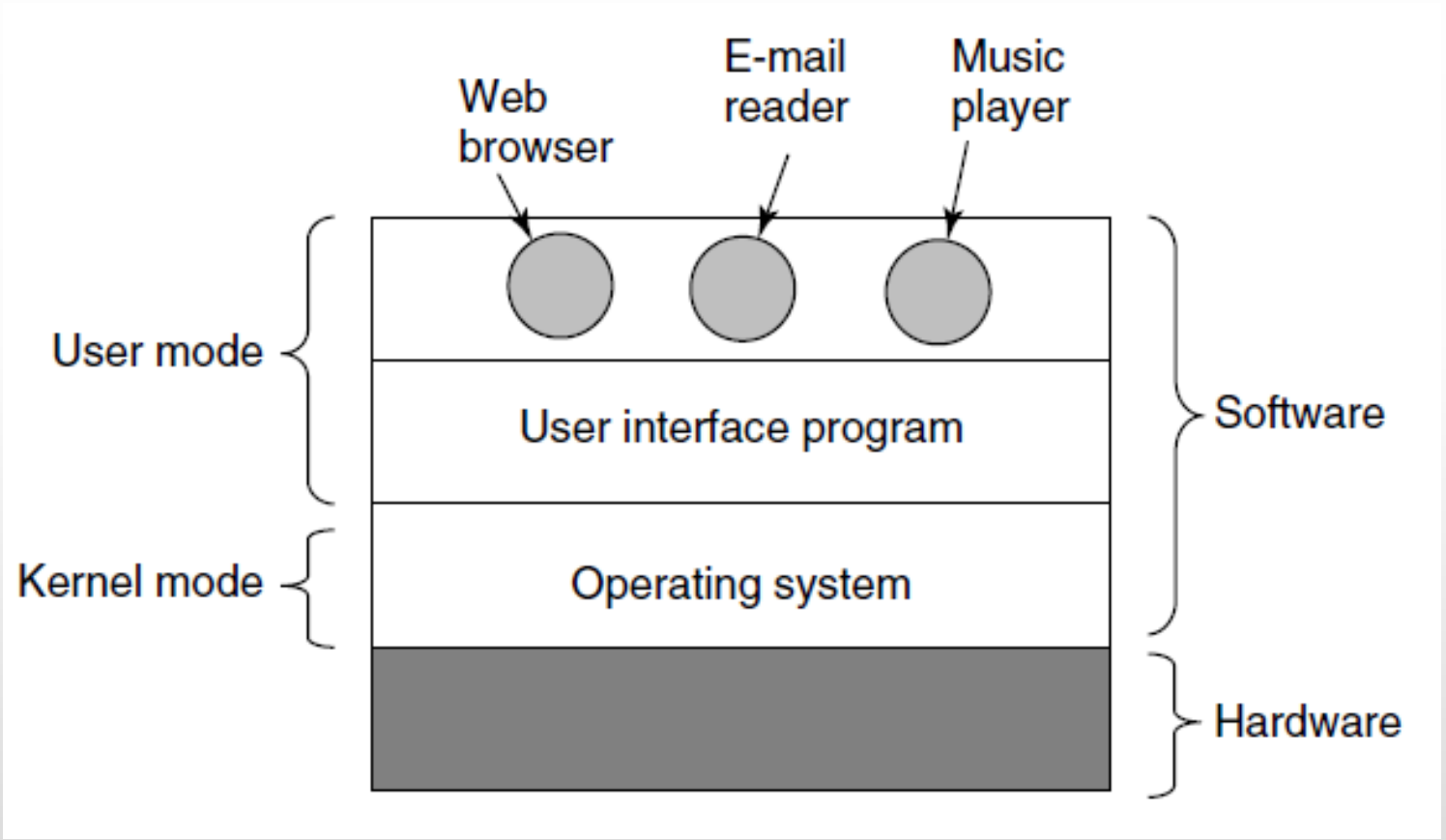
Aygıt Yöneticisi

- Aygıt yöneticisi, bilgisayara bağlı donanım aygıtlarını yönetmekten sorumlu işletim sisteminin bir bileşenidir.
- Cihazlar hakkında bilgi sağlar ve kullanıcının cihazları yapılandırmasına, güncellemesine ve sorunlarını gidermesine olanak tanır.
- Donanım aygıtlarının sorunsuz çalışması ve işletim sistemi ile uyumluluğunun sağlanmasında kilit rol oynar.
- Bilgisayarda çalışan ayrı bir uygulama veya program değildir.
- Cihaz türü, üreticisi ve sürümünün yanı sıra cihazla ilgili sorunlar veya problemler hakkında bilgi sağlar.

İşletim Sisteminin Konumu

- Bilgisayar donanımı ve yazılım arasında bir arayüzdür.
- Donanımın yazılım tarafından nasıl kullanılacağını yönetir.
- Donanım, fiziksel olarak mevcut olan bileşenleri (örneğin CPU, RAM, diskler) temsil eder.
- Yazılım ise, bilgisayarın yapabileceği işlemleri yürütmek için yazılmış kodları içerir.
- İşletim sistemi,
  - yazılımın donanımı kullanmasını kontrol eder,
  - donanımın kullanımını optimize eder ve
  - sistemin güvenliğini sağlar.

İşletim Sistemi Konumu



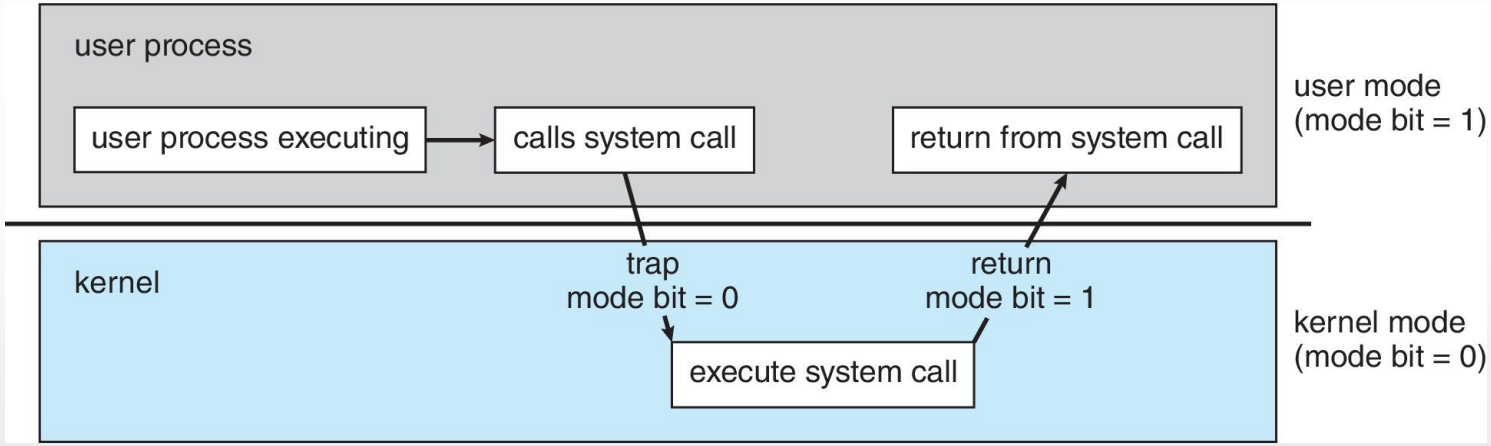
Çekirdek Modu ve Kullanıcı Modu

- Çoğu bilgisayarın iki çalışma modu vardır:

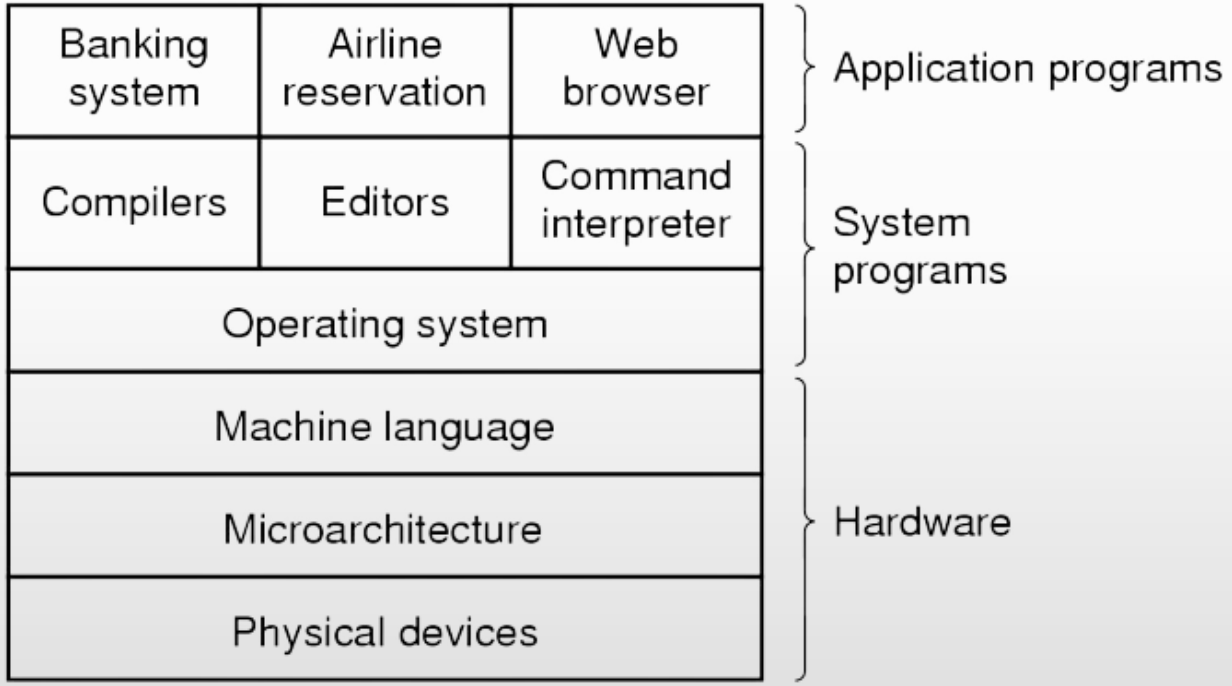
- İşletim sistemi, tüm donanıma tam erişime sahip olan ve herhangi bir talimatı yürütebilen çekirdek modunda çalışır.
- Yazılımın geri kalanı, sınırlı erişim ve kapasiteye sahip kullanıcı modunda çalışır.
- Kabuk ve GKA, kullanıcı modu yazılımının en düşük seviyesidir.

Çekirdek Modu ve Kullanıcı Modu

- .



İşletim Sistemi Konumu



Fiziksel Aygıtlar, Mikro Mimari, Makine Dili

- Fiziksel aygıtlar, bir bilgisayarın merkezi işlem birimi (CPU), bellek, depolama ve giriş/çıkış aygıtları gibi donanım bileşenleridir.
- Mikro mimari, veri yolu, bellek hiyerarşisi ve kontrol birimi dahil olmak üzere bilgisayarın işlemcisinin özel tasarımıdır.
- Makine dili, doğrudan bilgisayarın donanımı tarafından yürütülebilen ikili koddan oluşan en düşük seviyeli programlama dilidir.
- İşletim sistemi, aygıt sürücüleri aracılığıyla fiziksel aygıtlarla iletişim kurar ve bunları yönetir.
- Aygıt sürücüleri, üst düzey istekleri fiziksel aygıtlar tarafından yürütülebilecek düşük düzeyli komutlara çevirir.

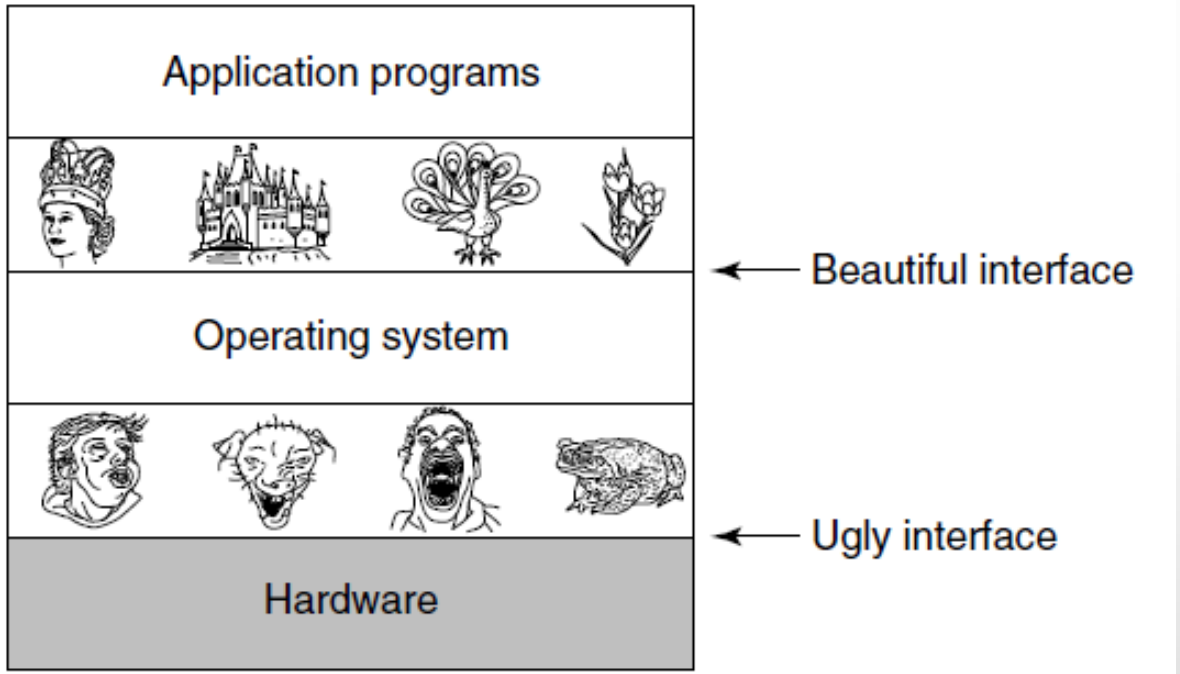
Fiziksel Aygıtlar, Mikro Mimari, Makine Dili

- İşletim sistemi, uygulamalar ile mikro mimari arasındaki arabirimi sağlayarak, uygulamaların bilgisayar kaynaklarına etkin bir şekilde erişmesine ve bunları kullanmasına izin verir.
- İşletim sistemi, üst düzey uygulamaları doğrudan bilgisayarın donanımı tarafından yürütülebilen makine diline çevirir.
- Donanım yapılandırması veya mikro mimarisi ne olursa olsun uygulamaların bilgisayarda yürütülmesine izin verir.
- İşletim sisteminin tasarımı ve uygulaması, donanım tarafından kullanılan makine dilinin yanı sıra bilgisayarın belirli mikro mimarisini ve fiziksel cihazlarını dikkate almalı ve hesaba katmalıdır.

Genişletilmiş Makine Olarak İşletim Sistemi

- Donanımın üstüne inşa edilmiş bir yazılımdır.
- Bilgisayar donanımını kullanmayı kolaylaştırır.
- Donanımın özelliklerini ve yeteneklerini kullanılabilir hale getirir.
- Donanımın özelliklerini gizler ve direkt kullanmasını engeller.
- İşletim sistemi ara yüzünü kullanmak daha kolaydır.
- İşletim sistemleri çirkin donanımları güzel soyutlamalara dönüştürür.

Genişletilmiş Makine Olarak İşletim Sistemi



Genişletilmiş Makine Olarak İşletim Sistemi

- Soyutlama:
  - İşlemci – Süreç
  - Depolama – Dosya
  - Bellek – Adres uzayı

- 4 tip çalışana ihtiyaç var:
  - Donanım tasarımcısı
  - Çekirdek tasarımcısı
  - Uygulama geliştirici
  - Son kullanıcı

Kaynak Yöneticisi Olarak İşletim Sistemi

- İşletim sistemi, bilgisayar donanımının kaynaklarını etkili bir şekilde yönetir. Kaynak kullanımını optimize eder. Kaynakların uygulamalar arasında adil bir şekilde dağıtımını sağlar.
- Üstten aşağıya bakış açısı:
  - Uygulama programları için soyutlamalar sağlar
- Aşağıdan yukarıya bakış açısı:
  - Karmaşık sistemin parçalarını yönetir
- Alternatif bakış açısı:
  - Kaynakların düzenli ve kontrollü dağıtımını sağlar.

Kaynak Yöneticisi Olarak İşletim Sistemi

- Birden çok programın aynı anda çalışmasına izin verir.
- Bellek, G/Ç cihazları ve diğer kaynakları yönetir ve korur.
- Kaynakları iki farklı şekilde paylaşır.
  - Zaman (time)
  - Alan (space)
- Birçok program aynı anda yazdırmak isterse ne olur?
- Her sürecin kaynak kullanımı/ihtiyacı nasıl hesaplanır?
- Kaynaklar çoklanırsa, adalet ve verimlilik nasıl sağlanır?

Birçok Program Aynı Anda Yazdırmak İsterse

- Programlar, yazdırma isteklerini, kaynakları adil ve verimli bir şekilde koordine eden ve yöneten işletim sistemine gönderir.
- İşletim sistemi, verimli ve etkili olmak için yazdırma isteklerini koordine eder ve önceliklendirir.
- İşletim sistemi, yazıcının kaynaklarını tahsis etmekten ve yazdırma isteklerinin yürütülme sırasını belirlemekten sorumludur.
- İşletim sistemi, eşzamanlı yazdırma isteklerini yönetmek için İlk Gelen, İlk Hizmet Alır (FCFS), Önce En Kısa İş (SJF), Round Robin ve Priority Scheduling gibi çeşitli zamanlama algoritmaları kullanır.
- Eşzamanlı yazdırma işlemini optimize etmek için, işletim sistemi ara belleğe alma ve biriktirme (spooling) tekniklerini kullanabilir.

Süreçlerin Kaynak Kullanımının Hesaplanması

- Bir sürecin kaynak kullanımı, yürütmek için gereken CPU zamanı, bellek ve G/Ç işlemleri gibi sistem kaynaklarının miktarı olarak tanımlanır.
- Windows'taki performans sayaçları ve Linux'taki /proc dosya sistemi gibi performans izleme araçları kullanılarak bir sürecin CPU zamanı kullanımı ölçülebilir.
- Bellek ayırma ve serbest bırakma modelleri izlenerek ve sürecin çalışma kümesinin boyutu izlenerek bellek kullanımı ölçülebilir.
- Sistem çağrıları ve G/Ç istekleri izlenerek gerçekleştirilen G/Ç işlemlerinin sayısı ölçülebilir.

Kaynaklar Çoklandığında Adalet ve Verimlilik

- Kaynaklardaki artış,
  - performans ve yanıt verebilirlik gibi faydalar sağlayabilir,
  - kaynakların adil ve verimli kullanımını sağlamak gibi zorluklar da getirir.
- İşletim sistemi adalet ve verimliliği dengelemek için kaynak zamanlama algoritmaları, öncelik sistemleri ve kaynak kotaları gibi yaklaşımlar kullanabilir.
- Kullanılacak yaklaşım,
  - sistemin ve kullanıcının özel gereksinimlerine ve hedeflerine ve
  - çalışan süreçlerin doğasına ve davranışına da bağlıdır.

İşletim Sistemlerinin Tarihi

- İlk jenerasyon (1945-55)
  - Vakum tüpleri
- İkinci jenerasyon (1955-65)
  - Transistörler ve toplu (batch) sistemler
- Üçüncü jenerasyon (1965-1980)
  - Entegre yongalar (IC) ve çoklu programlama
- Dördüncü jenerasyon (1980-günümüz)
  - Kişisel bilgisayarlar
- Beşinci jenerasyon (1990-günümüz)
  - Mobil bilgisayarlar

Vakum Tüpleri

- Büyük ve yavaş
- Mühendisler tasarlar, inşa eder, çalıştırır ve bakımını yapar
- Makine diliyle veya kablolar kullanılarak programlanır
- Takılabilir kartlar ile çalışır
- Ağırlıklı olarak sayısal hesaplamalar yapar

Transistörler ve Toplu (batch) Sistemleri

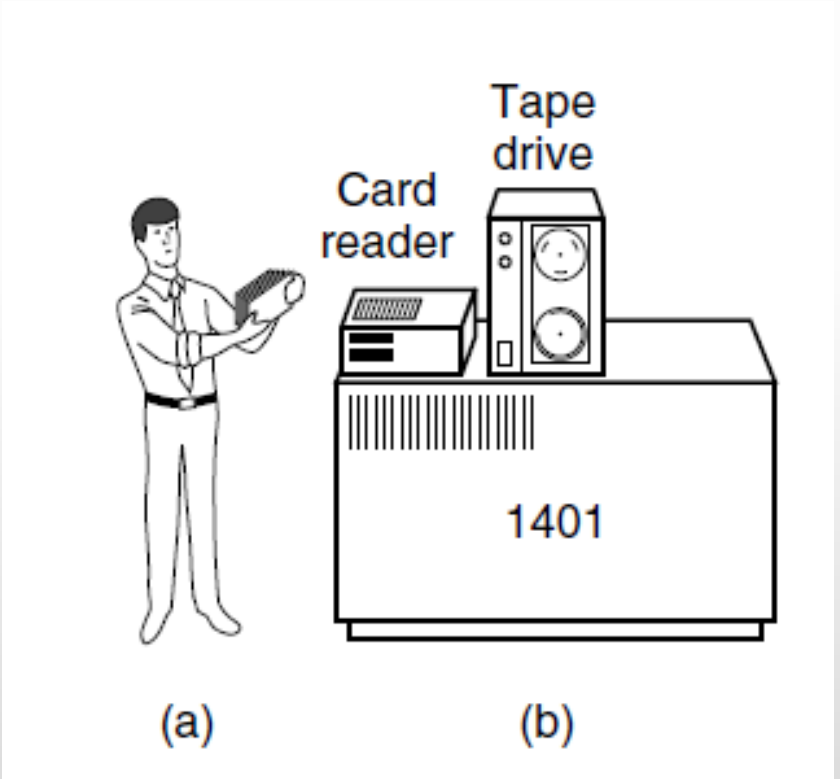
- Transistörlerin icadı ile birlikte ikinci jenerasyon işletim sistemleri ortaya çıktı. Transistörler, vakum tüplerin yerini aldılar
  - daha küçük, daha güvenilir ve daha enerji verimli
- Toplu sistemler, işlemlerin toplu olarak yürütülmesini sağlar. İşlemler işlem kuyruğuna eklenir ve işletim sistemi sırayla yürütür.
  - İşlemlerin paralel olarak yürütülmesini engeller.
  - İşlemlerin manuel olarak yürütülmesini gerektirir.
  - Veri işleme, hesaplama ve raporlama gibi işlemler için kullanılır
  - Gerçek zamanlı işlemler için uygun değildir.

Transistörler ve Toplu Sistemler

- Eski bir toplu sistem.

(a) Programcılar 1401'e kartlar getirir.

(b) 1401, iş kartlarını teyp'e okur.



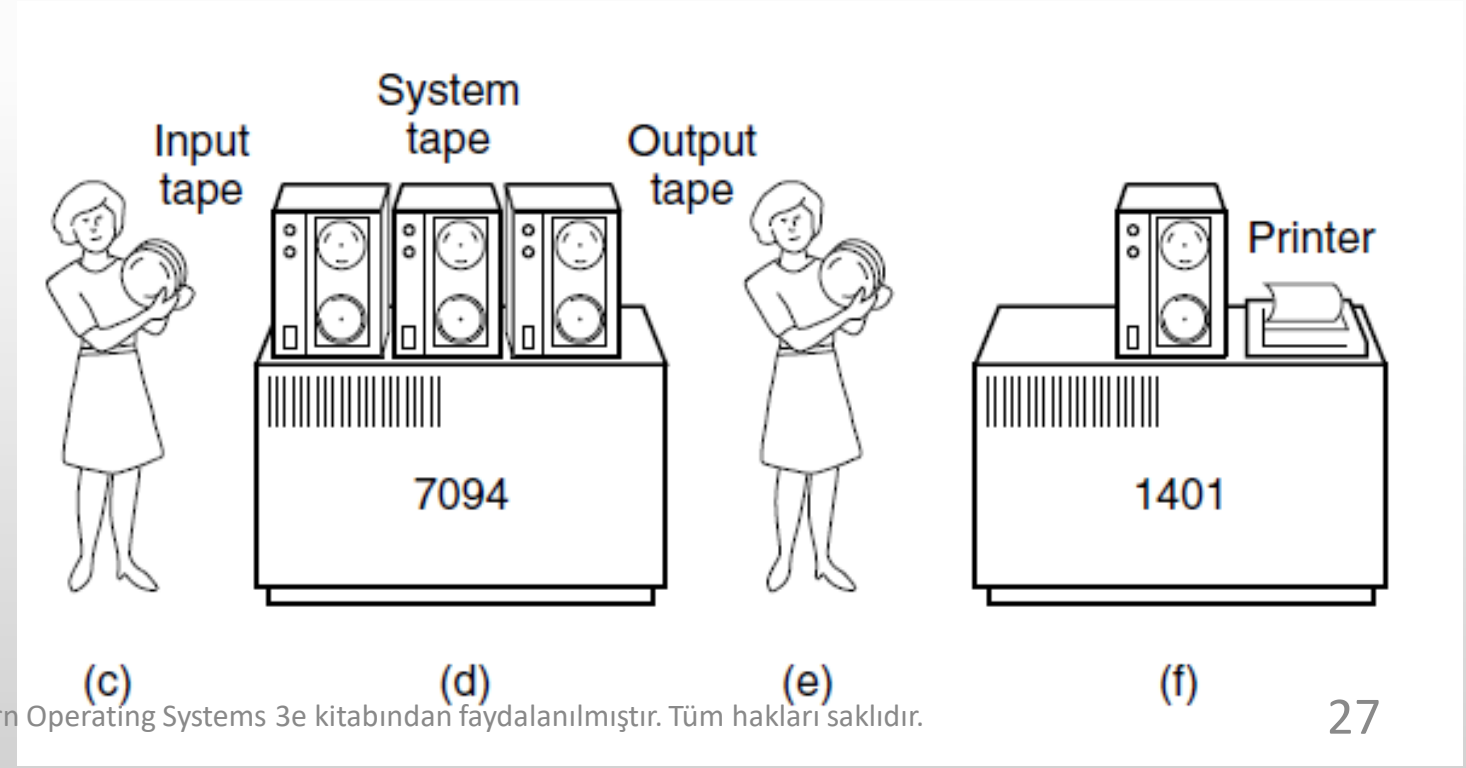
Transistörler ve Toplu Sistemler

(c) Girdi bandının 7094'e taşınması.

(d) 7094 hesaplamaları yapar.

(e) Çıktı bandının 1401'e taşınması.

(f) 1401 çıktıyı yazdırır.

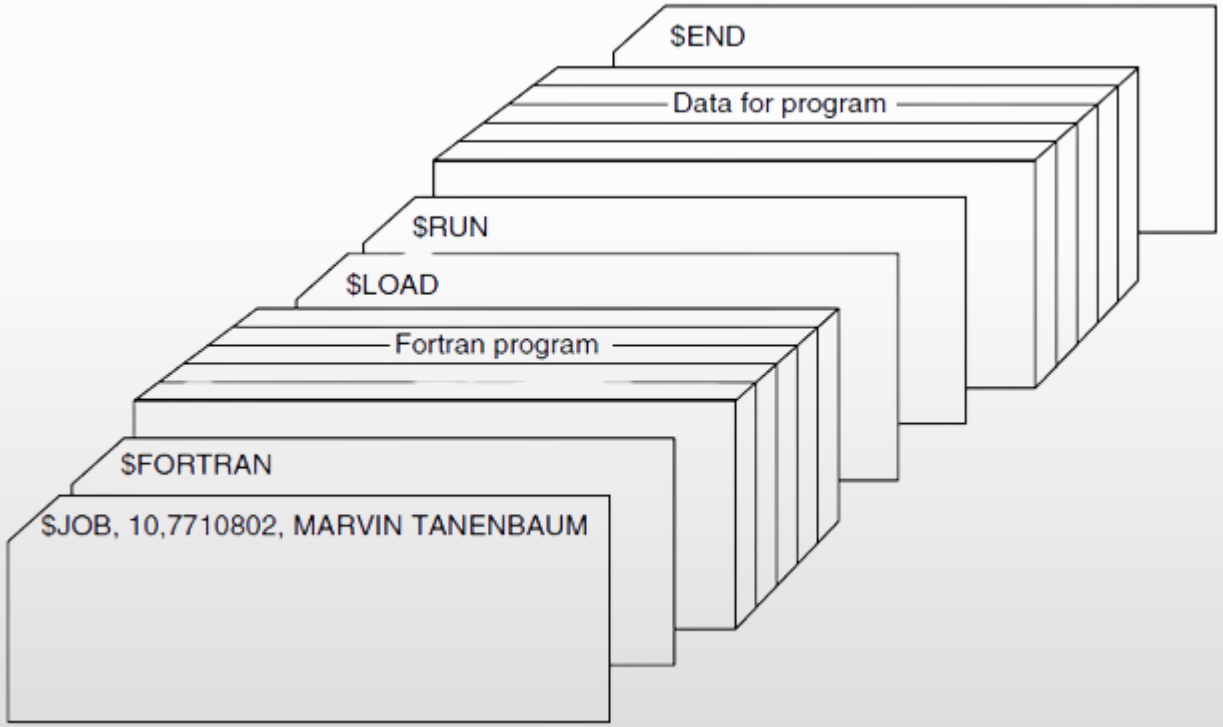


IBM 1401 ve 7094, System/360

- IBM 1401, 1959'da tanıtılan ticari ve bilimsel uygulamalarda yaygın olarak kullanılan ilk bilgisayarlardan.
- IBM 7094, 1962'de tanıtılan 1401'in daha güçlü bir versiyonu. Bilimsel ve teknik alanlarda kullanıldı. Zaman paylaşımını destekleyen ve çok kullanıcının aynı anda sisteme erişebildiği ilk bilgisayarlardan.
- Hem 1401 hem de 7094, ilk büyük ölçekli işletim sistemlerinden biri olan IBM System/360 İşletim Sistemi (OS/360) altında çalışıyordu.
- Birden çok uygulamayı aynı anda çalıştırmayı ve yazıcılar, teypler ve disk sürücüleri gibi kaynakları birden çok kullanıcı arasında paylaşmayı mümkün kıldı.

Tipik bir FMS İşinin Yapısı

- FMS (Flexible manufacturing system)



FMS Esnek Üretim Sistemi

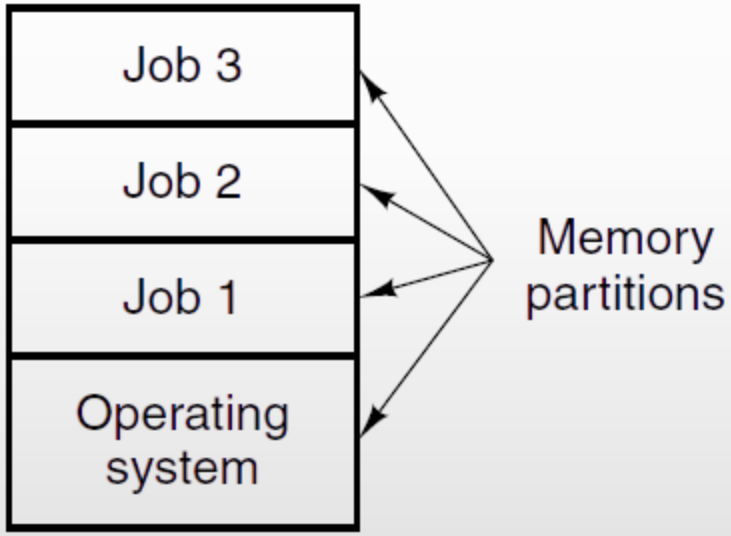
- FMS, bir fabrikadaki üretim sürecini otomatikleştiren bilgisayar kontrollü bir sistemdir.
- Malzeme taşıma, işleme ve montaj gibi bir dizi üretim sürecini tek bir otomatik sistemde birleştirir.
- İşletim sistemleri, FMS'nin çalışmasında çok önemli bir rol oynamaktadır.
- FMS ortamında işletim sistemi, güvenilir, verimli, ve ölçeklenebilir olmalıdır.
- FMS ortamlarında özel işletim sistemlerinin kullanılması,
  - üretim verimliliği,
  - gerçek zamanlı kontrol,
  - geliştirilmiş veri yönetimi sağlar.

Bütünleşik Devreler ve Çoklu Programlama

- Bütünleşik devrelerin (IC) icadı ile birlikte üçüncü jenerasyon işletim sistemleri ortaya çıktı. Devreler transistörlerin yerini aldı.
  - Daha küçük, daha güvenilir ve daha enerji verimli
- Çoklu programlama, birden fazla işlemi aynı anda yürütmek için kullanılır.
  - Dinamik olarak işlemlerin ağırlıklarının ayarlanmasını sağlar.
  - İşlemler arasında eşitliği sağlar
  - İşlemlerin paralel olarak yürütülmesini sağlar.
  - Gerçek zamanlı işlemler için uygun.

Bütünleşik Devreler ve Çoklu Programlama

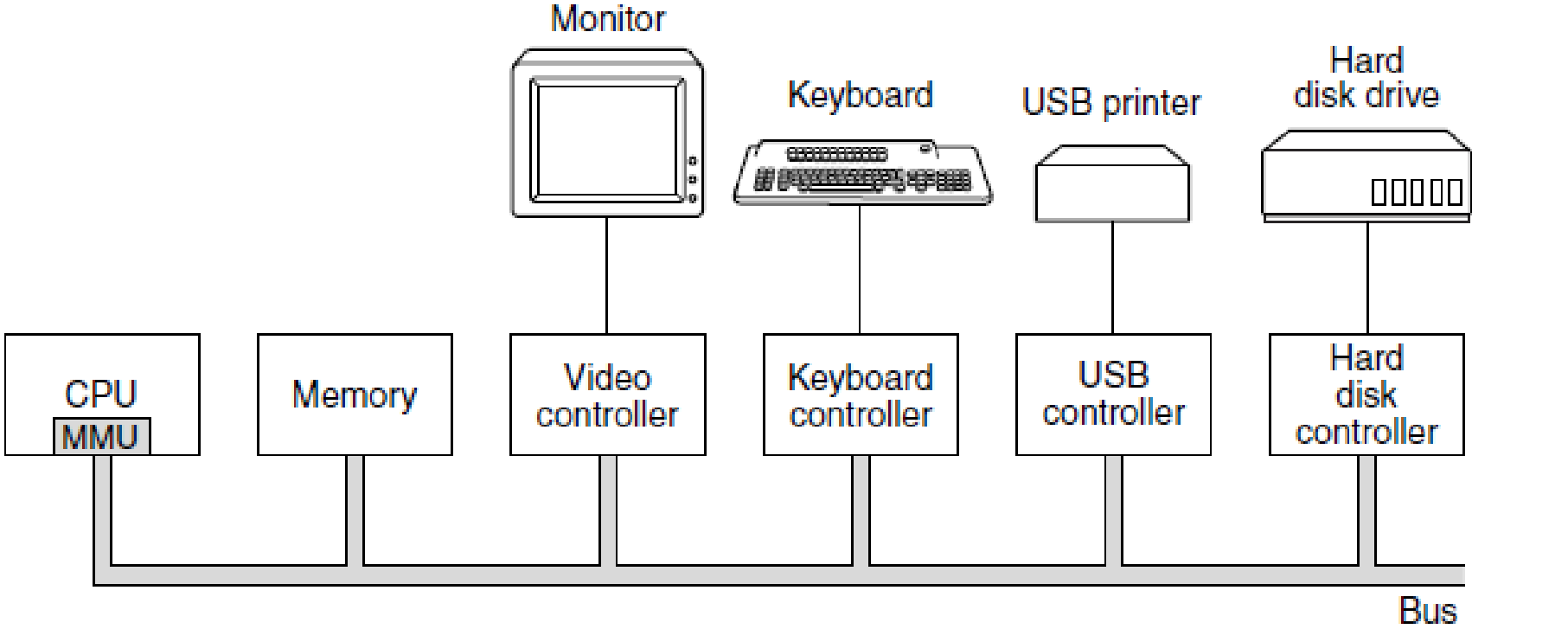
- Bellekte üç işi olan bir çoklu programlama sistemi.



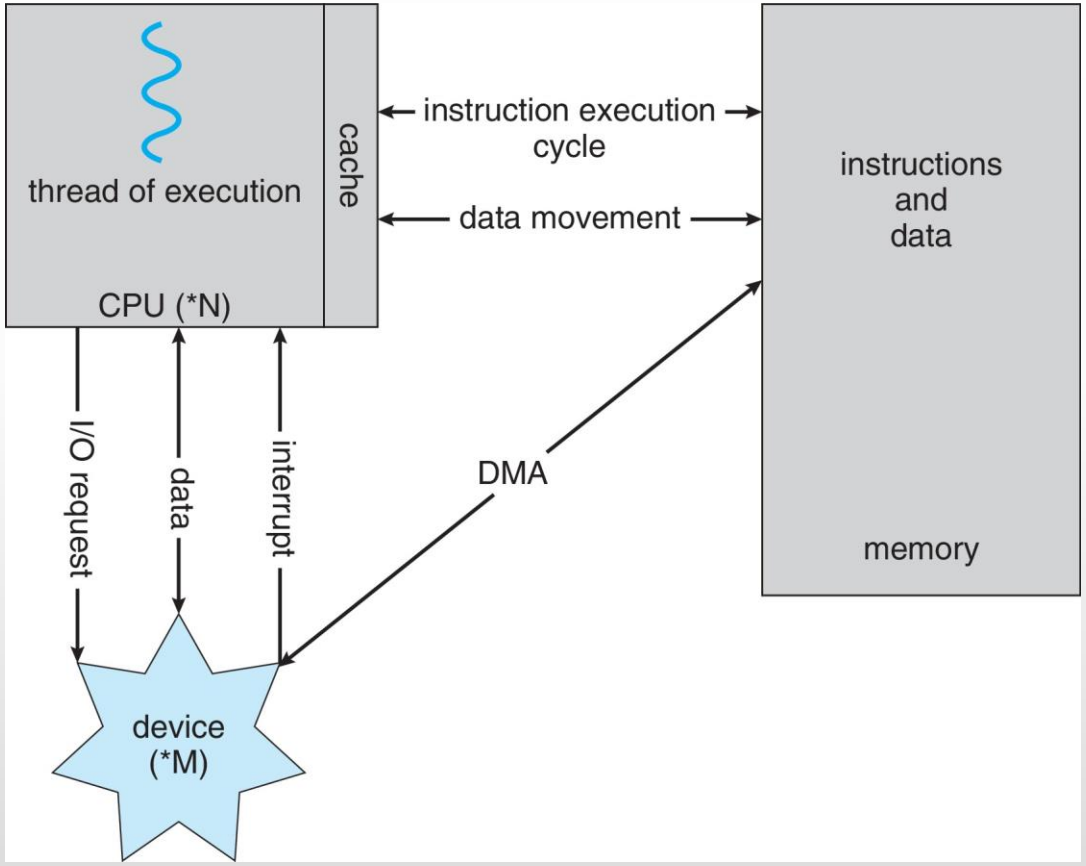
Kişisel Bilgisayarlar

- Bilgisayarlar üçüncü nesil benzeri performansa sahiptir, ancak fiyatları büyük ölçüde düşmüştür
- CP/M: İlk disk tabanlı işletim sistemi
- 1980, IBM PC, Basic Interpreter, DOS, MS-DOS
- GUI, Lisa, Apple: kullanıcı dostu
- Grafik arayüzü MS-DOS, Win95/98/ME, winNT/XP

Kişisel Bilgisayarın Bazı Bileşenleri



Von Neumann Mimarisi



İşlemciler

- Bilgisayarın en önemli bileşenidir
- Tüm işlemleri yürütmek için kullanılır.
- İşlemci,
  - Bilgisayar kodunu anlar
  - Kodu yürütmek için gerekli olan işlemleri gerçekleştirir.
  - Çok çekirdekli yapıda olabilir
  - Birden fazla işlemi aynı anda yürütebilir.
  - Farklı hız, çekirdek sayısı, önbellek boyutu, veri yolu genişliği gibi özelliklere sahip olabilir.

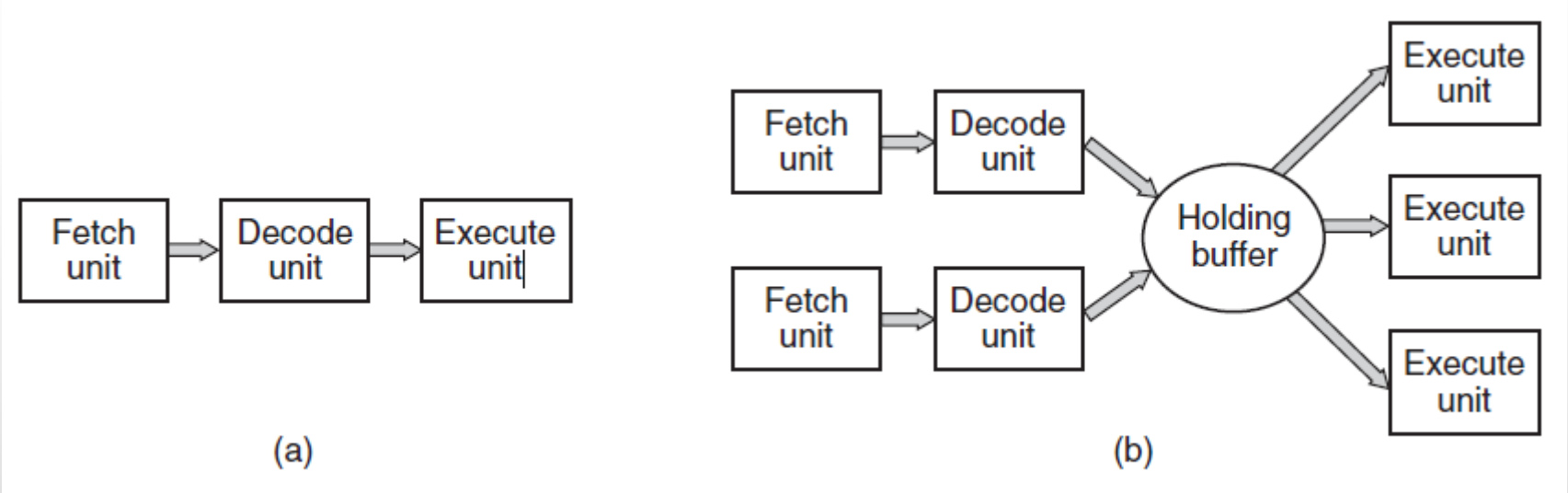
İşlemciler

- Bilgisayarın beyni
- Komutu bellekten alır ve yürütür
- CPU Döngüsü:
  - Getir (fetch), kodunu çöz (decode), yürüt (execute)
- CPU, değişken ve geçici sonuçları saklamak için yazmaçlara sahiptir:
  - Bellekten yazmaca yükle
  - Yazmaçtan belleğe sakla
- Program sayacı: işletilecek bir sonraki komut
- Yığıt işaretçisi: geçerli yığıtın en üstü
- PSW: program durum sözcüğü, öncelik, mod, …

İşlemciler

(a) Üç aşamalı bir boru hattı (pipeline).

(b) Bir superscalar CPU



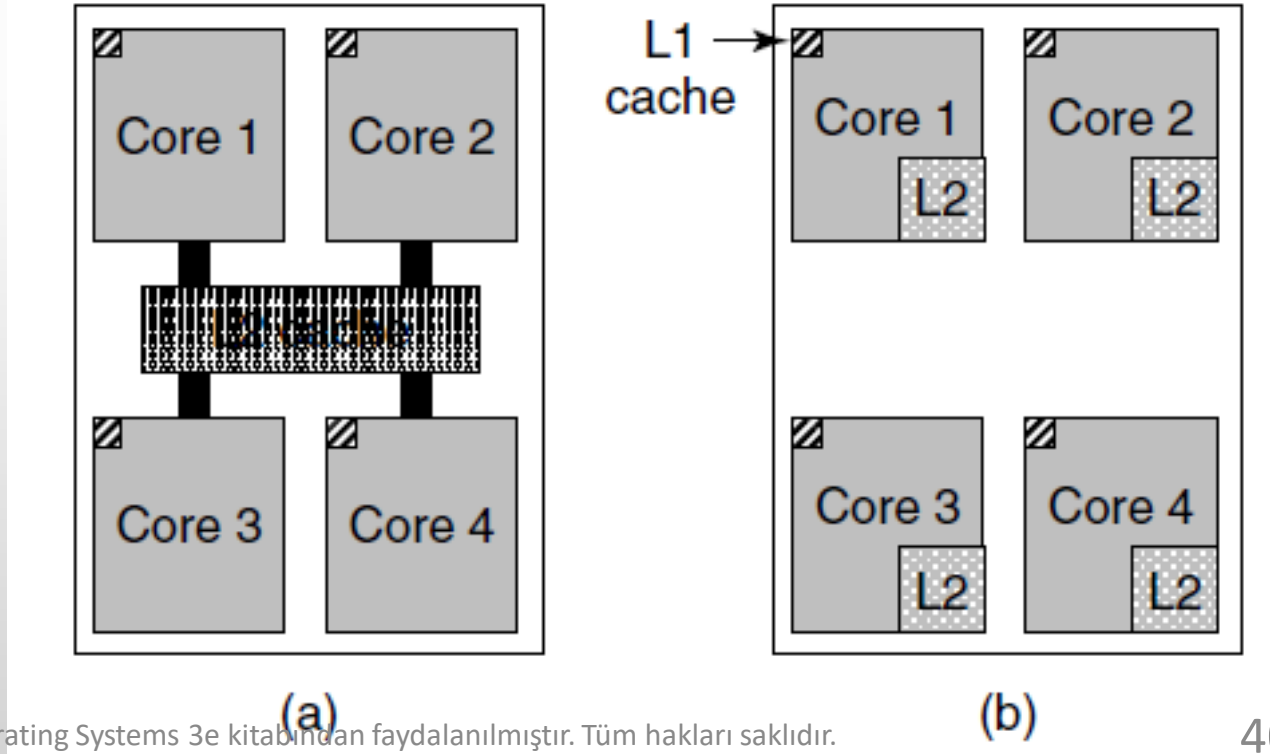
Bellek

- İşlemciler tarafından okunabilecek ve yazılabilecek verileri geçici olarak saklamak için kullanılan bileşendir.
- RAM (Random Access Memory) olarak da adlandırılır.
- Bellek boyutu, bilgisayarın performansını ve kullanılabilirliğini etkiler.
- Bellek, işlemler arasında verileri paylaşmayı ve hızlı erişimi sağlar.

Bellek

(a) Paylaşımlı L2 önbellekli bir dört çekirdekli chip.

(b) Ayrı L2 önbellekli dört çekirdekli chip.



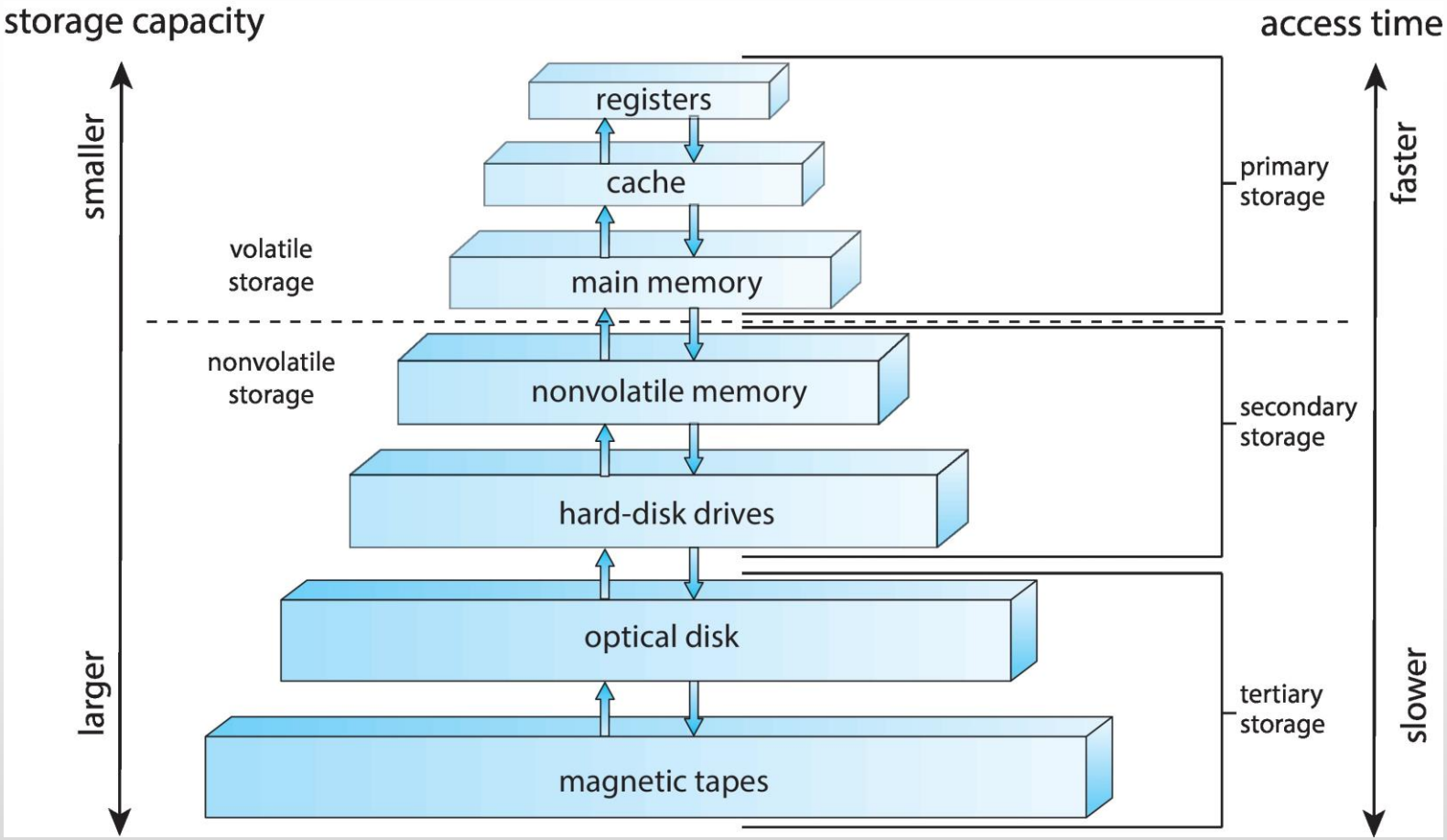
Önbellek (cache)

- Önbellekler, işlemci tarafından sıklıkla erişilen verileri geçici olarak depolamak için kullanılan yüksek hızlı bellek birimleridir.
- Verilere erişmek için hızlı, düşük gecikmeli bir yol sağlar.
- L1 önbellek: En küçük ve en hızlı önbellek türüdür ve genellikle doğrudan işlemciye entegre edilir.
- L2 önbellek: Genellikle işlemci ile aynı çipte bulunan daha büyük, daha yavaş bir önbellektir.
- L3 önbellek: En büyük ve en yavaş önbellek türüdür ve genellikle çip dışında bulunur.

Ana Bellek

- RAM, (random access memory)
  - Değiştirilebilir, hızlı, pahalı
- ROM, (read only memory)
  - Değiştirilemez, hızlı, ucuz
  - BIOS, İşletim sistemi yükleyici ..
- EEPROM (Electrically Erasable Programmable ROM)
  - Yeniden yazılabilir, yavaş
  - Taşınabilir müzik oynatıcılarındaki diskler ..

Bellek Hiyerarşisi



Bellek Hiyerarşisi

Level	1	2	3	4	5
Name	registers	cache	main memory	solid-state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25-0.5	0.5-25	80-250	25,000-50,000	5,000,000
Bandwidth (MB/sec)	20,000-100,000	5,000-10,000	1,000-5,000	500	20-150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

Bellek

- Bellek, bilgisayarın verileri alıp depoladığı yerdir
- İdeal olarak, çip şeklinde ve büyük olmalıdır
- Bellek hiyerarşisi göz önünde bulundurulmalıdır
- Önbellek satırları:
  - Bellek, önbellek satırlarına bölünür; en çok kullanılanlar önbellekte saklanır
  - Cache hit/miss, aranan verinin önbellekte olup/olmaması
  - Performansı artırmak için kullanılır

Önbellek

- Ana bellek, önbellek satırlarına bölünmüştür (64 bayt)
  - 1. satırda 0-63, 2. satırda 64-127
- Program bir sözcük (word) okuduğunda, donanım önbellekte olup olmadığını kontrol eder.
  - Önbellekte ise, cache hit olur (2 döngü cycle)
  - Değilse, veri yolu üzerinden ana bellekten talep et (maliyetli)
- Önbellek pahalı olduğundan boyutu sınırlıdır
- Önbellek hiyerarşilere sahip olabilir

Önbellek Sorunları

- Yeni bir öğe önbelleğe ne zaman yerleştirilmeli?
- Yeni öğe hangi önbellek satırına koyulmalı?
- Yer açmak için önbellekten hangi öğe çıkarılmalı?
- Çıkarılan öğe bellekte nereye yerleştirilmeli?

Önbellek Sorunları

- Önbelleği yönetmede çeşitli faktörler göz önünde bulundurulur:
- **Önbellek boyutu**: Ne kadar veri depolanabileceğini ve dolayısıyla performansın ne kadar iyileştirilebileceğini etkiler.
- **Önbellek değiştirme ilkesi**: İşletim sistemi, önbellek dolduğunda hangi verilerin değiştirileceğine karar vermeli ve performans ile verimliliği dengeleyen etkili bir değiştirme ilkesi seçmelidir.
- **Önbellek tutarlılığı**: Birden çok işlemci kullanıldığında, eşzamanlı erişimlerin varlığında bile önbellek tutarlı ve güncel tutulmalıdır.

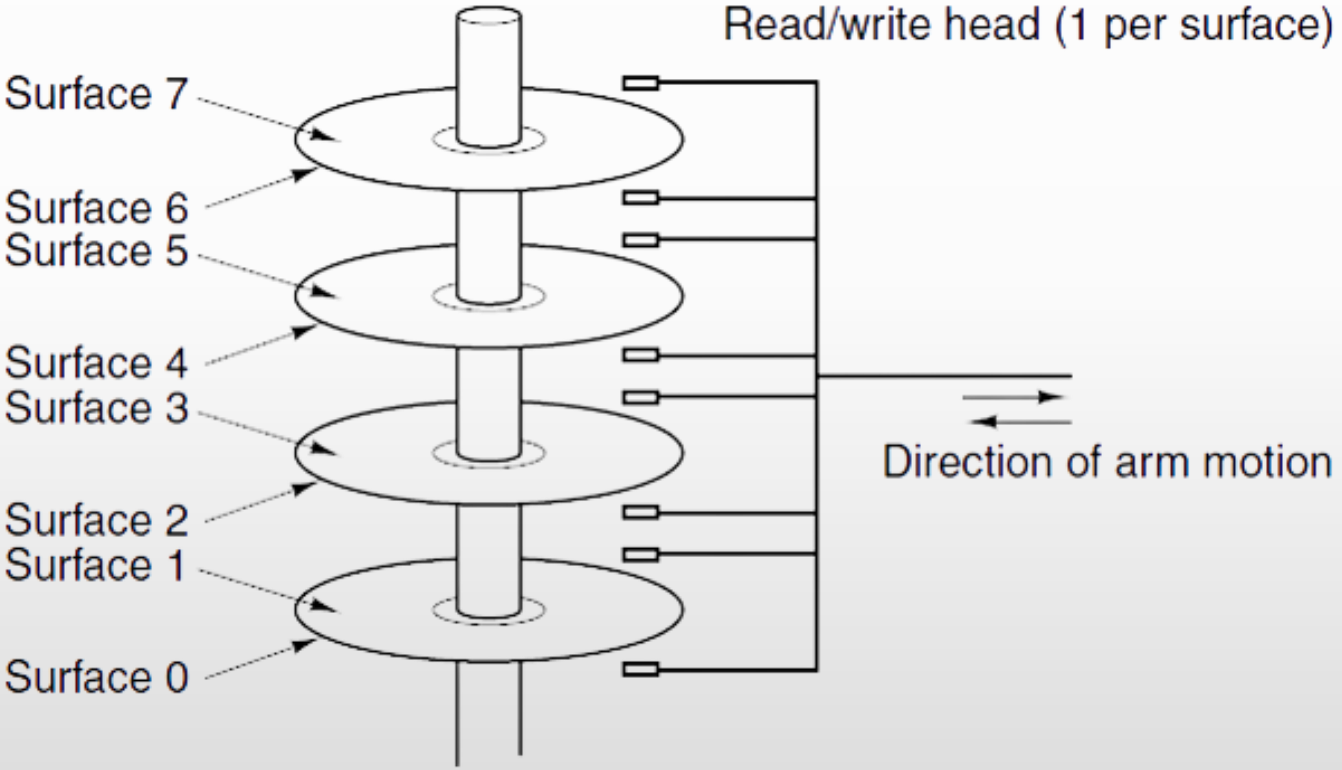
Disk

- Verileri uzun vadeli saklamak için kullanılan cihazlardır.
- Okuma ve yazma işlemleri için veriler disk plakaları üzerinde saklanır.
- Disk sürücüleri, farklı boyutlarda ve kapasitelerde olabilir.
- Disk sürücüsü yapısı, disk plakası, okuyucu/yazıcı kafası, motor ve kontrol elemanlarından oluşur.
  - Disk plakası, verileri saklamak için kullanılan alandır.
  - Okuyucu/yazıcı kafası, verileri okuma ve yazma işlemleri için kullanılır.
  - Motor, disk plakasını döndürür ve yazıcı kafasını hareket ettirir.
  - Kontrol elemanları, disk sürücüsünün işlemlerini yönetir.

Disk

- Ucuz, büyük, ancak yavaştır. Mekanik hareketlere ihtiyaç duyar.
- Arm, track, cylinder, sector, head, checksum
  - x diskinde y sektörünü oku komutunu alır.
  - x ve y bilgisini [cylinder, sector, head] adres şekline çevirir.
  - Kolu doğru silindire hareket ettirir.
  - Kafanın doğru sektör üzerine gelmesini bekler.
  - Sürücüden gelen bitleri okur ve saklar.
  - Sağlama yapar. Okunan bitleri sözcük olarak bellekte saklar.
- Disk, Sanal belleğin uygulanmasına yardımcı olur
  - Bellek yetersiz kaldığında, depolama alanı olarak diskler kullanılır.

Disk Sürücüsünün Yapısı



G/Ç Cihazları

- Bilgisayarın veri alma/gönderme işlemlerini gerçekleştiren cihazlardır.
- Dış dünya ile bilgisayar arasındaki veri transferini sağlar.
- Çeşitli tipte olabilir: Klavye, fare, ekran, yazıcı, tarayıcı, ses kartı, kameralar, vb.
- İşletim sistemi tarafından yönetilir ve kullanıcının cihazları kullanmasına izin verir.
- Bilgisayarın performansını ve kullanılabilirliğini etkiler.

G/Ç Cihazları

- İki parça: bir denetleyici ve bir aygıt
- **Denetleyici**: işletim sistemine daha basit bir arayüz sağlar
- **Aygıt sürücüsü**: denetleyiciyle konuşur, komut verir ve yanıt alır
- İletişim modları:
  - Meşgul bekleme
  - Kesme
  - DMA

Aygıt Sürücüsü

- İşletim sistemi denetleyiciyle konuşur. (komut verir, yanıt alır)
- Denetleyici üreticileri, her işletim sistemi için bir sürücü sağlar
- Sürücü, çekirdek modunda çalışır
- Denetleyici, sürücüyle iletişim kurmak için yazmaçlar kullanır.
- Üç iletişim modu
  - Sorgulama (polling)
  - Kesmeler (interrupt)
  - DMA

Sorgulama, Kesilme ve Direkt Bellek Erişimi

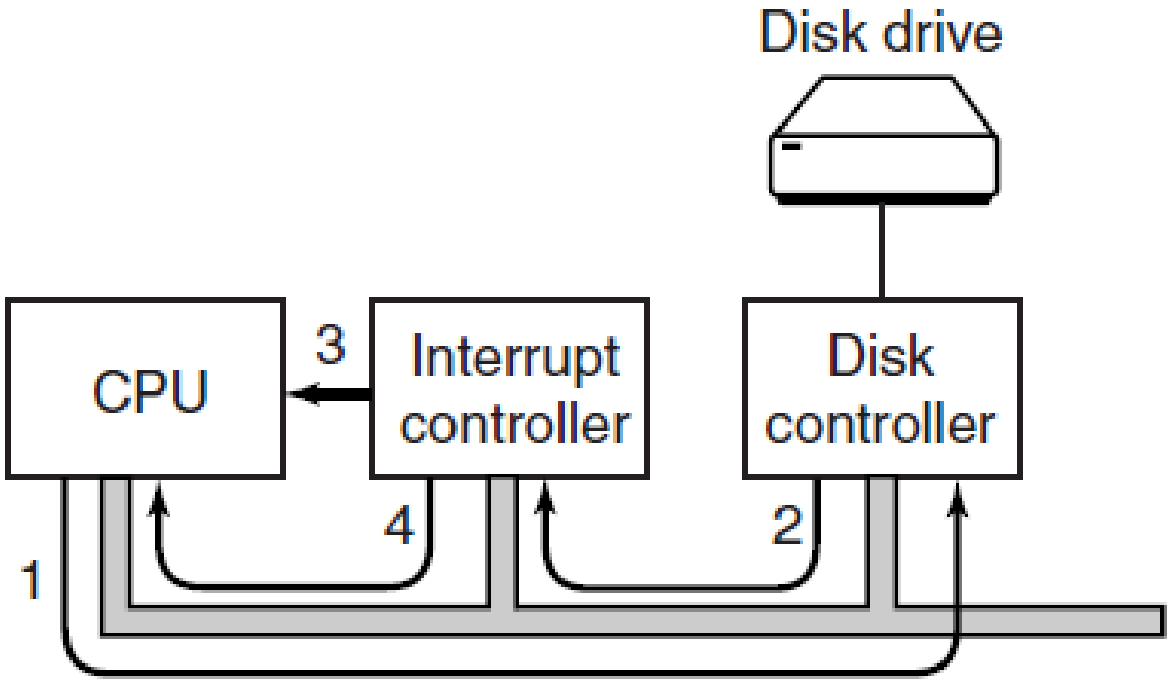
- **Sorgulama** (polling), aygıtın hazır olup olmadığını görmek için aygıtın durumu kontrol edilir. İşletim sistemi, bir döngü veya zamanlayıcı kullanarak aygıtı sürekli olarak denetler.
- **Kesilmeler** (interrupt), aygıtın ilgilenilmesi gerektiğini işletim sistemine bildirme yöntemidir. Bir aygıt bir kesme oluşturduğunda, işletim sistemine bir sinyal gönderir, işletim sistemi mevcut görevini durdurur ve kesmeyi işler. Sorgulamadan daha verimli ve duyarlı bir yöntemdir.
- **DMA**, işlemciyi dahil etmeden doğrudan bir cihaz ile ana bellek arasında veri aktarma yöntemidir. DMA, büyük miktarda veri aktarımı için daha hızlı ve daha verimli bir yöntem sağlar.

G/Ç Cihazları - Sorgulama

- Sürücü, denetleyiciye komut verir
- Sürücü, aygıt hazır olana kadar sorgular
  - Örneğin, kabul etmeye hazır olana kadar yazıcı denetleyicisini sorgula, karakter gönder
- Yüksek CPU kullanımına neden olur
- Programlanmış G/Ç olarak adlandırılır, artık kullanılmıyor

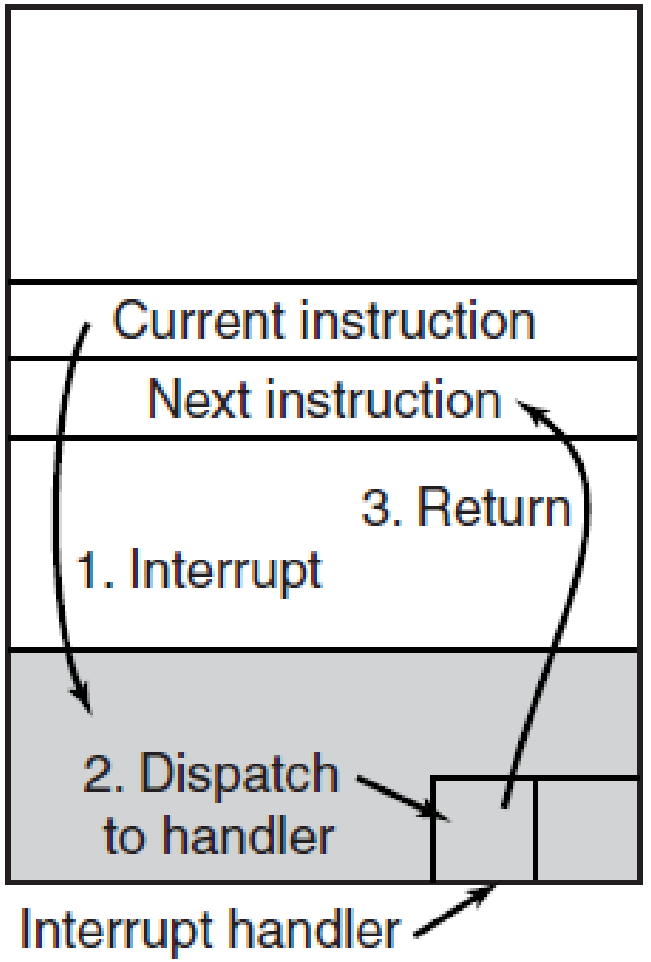
G/Ç Cihazları - Kesme

G/Ç cihazını başlatma ve bir kesme alma adımları



G/Ç Cihazları - Kesme

- Kesme işleme,
  - kesmeyi alma
  - kesme işleyicisini çalıştırma
  - kullanıcı programına dönme
- G/Ç işlemi bittiğinde kesme üret
- İşlem yapılırken işlemcinin başka işler yapmasına izin verir.



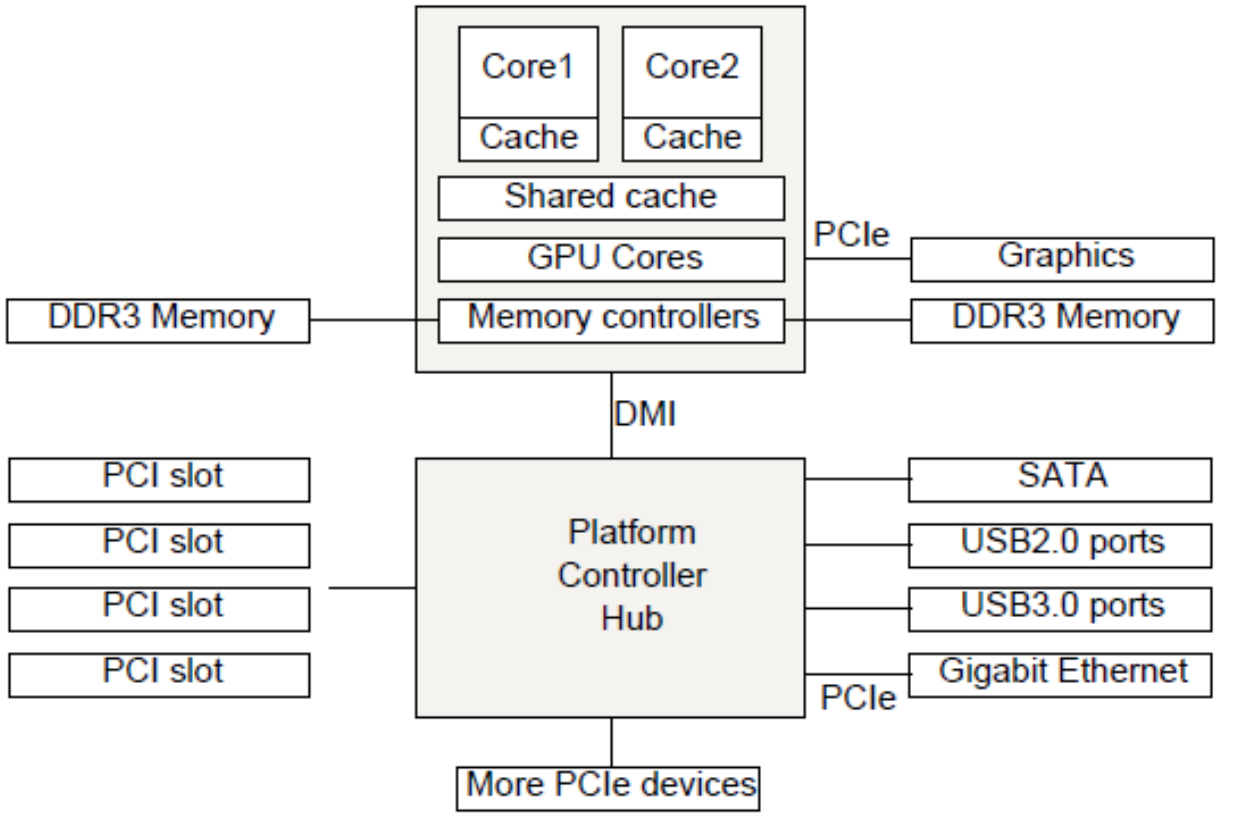
G/Ç Cihazları - DMA

- Özel (denetleyici) yonga var
- Bellek ile veri transferinde işlemci kullanmaktan kaçınır
- İşlemci, yongaya aktarım hakkında gerekli bilgileri verir
- Yonga işlem bittiğinde kesme üretir.

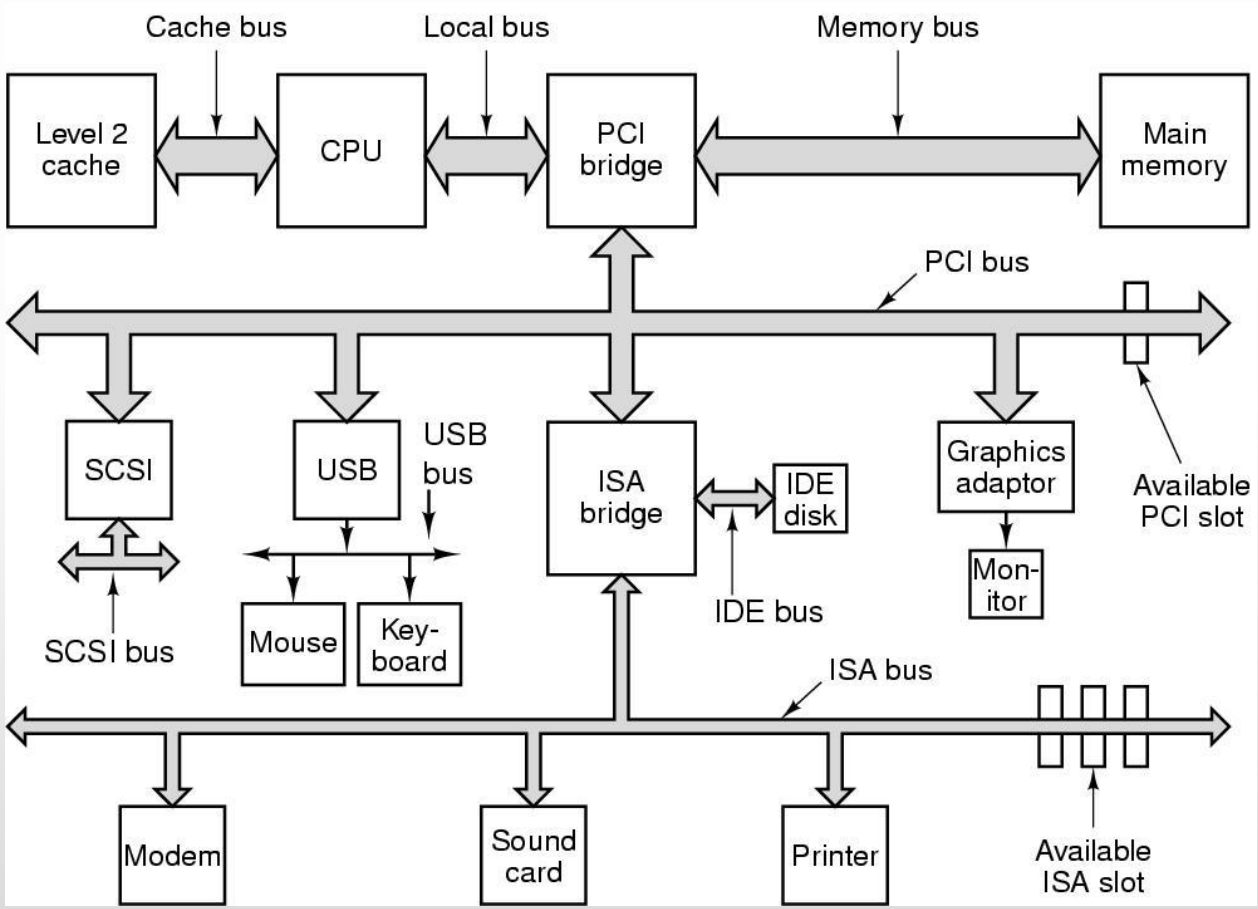
Veriyolları

- Veriyolları, bilgisayar bileşenleri arasında veri ve sinyallerin taşınması için kullanılır.
  - Örneğin, işlemci, bellek, G/Ç cihazları arasında veri taşır.
- Bant genişliği ve hızı açısından değişebilir.
  - Örneğin, PCI, PCI-Express, USB gibi.
- Veri taşınma yönetiminden işletim sistemi sorumludur.
- Bilgisayarın performansını ve kullanılabilirliğini etkiler.
- Eskiden bir veri yolu vardı, yetmeyince daha hızlı (PCI), özelleştirilmiş (SCSI, USB) veri yolları çıktı.

X86 Sistem Yapısı



Pentium Sistem Veriyolları



Veriyolları

- PCI, ağ kartları, ses kartları ve grafik kartları gibi çevre birimlerini ana karta bağlayan yüksek hızlı bir yerel veri yoludur. Sisteme yeni çevre birimleri eklemek için esnek ve ölçeklenebilir bir çözüm sunar.
- ISA, eskiden yaygın olarak kullanılan, PCI'ye kıyasla daha yavaş ve sınırlı bir veri yolu standardıdır.
- SCSI, sabit sürücüler ve teypler gibi depolama aygıtlarını bağlamak için kullanılır. Tek bir veri yolunda birden çok depolama aygıtını bağlayan hızlı ve esnek bir çözüm sunar.
- USB, klavye, fare ve flash sürücüler gibi çevre birimlerini bağlamak için yaygın olarak kullanılan yüksek hızlı bir veri yoludur.
- IDE, sabit sürücüler ve CD-ROM gibi depolama aygıtlarını bağlamak için yaygın olarak kullanılır. Basit ve uygun maliyetli bir çözüm sunar.

Bilgisayarın Ayağa Kalkması

- BIOS: temel giriş/çıkış sistemi
- Ana kartta yer alır, düşük seviye G/Ç yazılımı

1. Bellek, klavye ve diğer temel cihazları kontrol eder
2. Önyükleme aygıtını belirler (disket, CD-ROM, disk)
3. Önyükleme aygıtının ilk sektörü belleğe okunur

- Sektör, hangi bölümün aktif olduğunu kontrol etmek için program içerir

4. Ardından, ikincil bir önyükleyici belleğe okunur
5. İşletim sistemi aktif bölümden okunur.

İşletim Sistemi Çeşitleri

- Anabilgisayar (mainframe)
- Sunucu (server)
- Çoklu işlemci (multiprocessor)
- Kişisel (personal)
- Mobil (handheld)
- Gömülü (embedded)
- Algılayıcı düğüm (sensor node)
- Gerçek zamanlı (real-time)
- Akıllı kart (smart card)

Anabilgisayar (mainframe) İşletim Sistemi

- Büyük ve karmaşık bilgisayar sistemleri için tasarlanmıştır.
- Çoklu kullanıcı ve çoklu işlem desteği sunar.
- Yüksek performans, yüksek güvenilirlik ve yüksek kullanılabilirlik sunar.
- Büyük veri setleri ve yüksek trafikli işlemler için kullanılır.
- Endüstriyel ve kurumsal uygulamalar için kullanılır.
- IBM z/OS, Unisys MCP, Fujitsu BS2000/OSD ...

Sunucu (server) İşletim Sistemi

- Çoklu kullanıcı, çoklu işlem ve yüksek kullanılabilirlik için optimize edilmiştir.
- Sunucu cihazlarında veri depolama, dosya paylaşımı, veritabanı işlemleri, web sunucusu hizmetleri ve diğer hizmetleri sağlar.
- Sunucu işletim sistemleri, kurumsal ve endüstriyel ortamlarda yaygın olarak kullanılır.
- Kuruluşların veri merkezleri, bulut bilişim ve diğer hizmetleri sağlamak için kullanılır.
- Windows Server, Linux, UNIX ...

Çoklu İşlemci (multiprocessor) İşletim Sistemi

- Birden fazla işlemciye sahip bilgisayarlarda paralel işlemleri gerçekleştirmek için kullanılır.
- İşlemlerin işlemci üzerinde aynı anda çalışmasını sağlar ve bu sayede işlemlerin hızını arttırır.
- Linux, UNIX, Windows ...

Kişisel (personal) İşletim Sistemi

- Ev kullanıcıları, öğrenciler ve küçük işletmeler için tasarlanmıştır.
- Kullanıcıların çeşitli uygulamaları ve yazılımları yüklemek, internette gezinmek ve dosyaları yönetmek gibi işlemleri gerçekleştirmek için kullanılır.
- Kullanıcı dostu arayüzleri ve kolay kullanımı sunar.
- Windows, MacOS, Linux ...

Mobil (handheld) İşletim Sistemi

- Taşınabilir cihazlar için tasarlanmıştır. (akıllı telefonlar, tabletler)
- İnternet erişimi, e-posta, sosyal medya, navigasyon, müzik ve video oynatma gibi hizmetler sağlar.
- iOS, Android, Windows Phone ...

Gömülü (embedded) İşletim Sistemi

- Otomatikleştirilmiş sistemler, cep telefonları, ev otomasyonu, araba sistemleri, hava taşıtları gibi cihazlar için tasarlanmıştır.
- Sistem ve cihazların özelliklerini optimize etmek için tasarlanmıştır.
- Linux, VxWorks, QNX ...

Algılayıcı düğüm (sensor node) İşletim Sistemi

- Algılayıcı, IoT, M2M ağları için tasarlanmıştır.
- Algılayıcı verilerini toplamak, işlemek ve iletmek için kullanılır.
- Enerji verimliliği ve güç tüketimi için optimize edilmiştir.
- TinyOS, Contiki, RIOT ...

Gerçek Zamanlı (real-time) İşletim Sistemi

- Zaman kritik işlemlerin zamanında yerine getirilmesi için gerçek zamanlı uygulamalar tarafından kullanılır.
- Ses, video, hareket ve diğer algılayıcı verilerini işlemek için kullanılır.
- Tahmin ve kontrol uygulamaları, otomatik sistemler, tren kontrol sistemleri, askeri araçlar gibi alanlarda kullanılır.
- VxWorks, QNX, RTLinux ...

Akıllı Kart (smart card) İşletim Sistemi

- Akıllı kartlar, küçük boyutlu ve güvenli cihazlar için tasarlanmıştır.
- Kimlik doğrulama, para transferi, elektronik para, kriptografik işlemler ve güvenli veri depolama için kullanılır.
- Kredi kartları, banka kartları, yolcu uçuş kartları, kimlik kartları ve diğer kartlar için kullanılır.
- JavaCard, MULTOS …

Zamanlayıcı (timer)

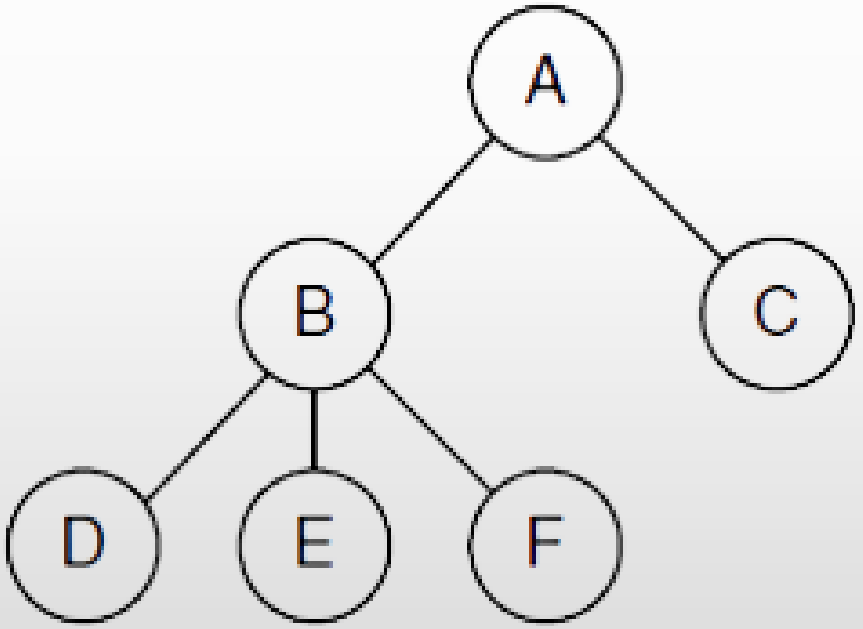
- Sonsuz döngüyü önlemek için kullanılır
- Belirli bir süre sonra kesilme yaratır
  - Fiziksel saat (clock) tarafından azaltılan bir sayaç tutulur
  - İşletim sistemi sayacı ayarlar (ayrıcalıklı talimat)
  - Sayaç sıfır olduğunda kesilme üretilir
  - İşletim sistemi kontrolü tekrar kazanır
  - Kendisine ayrılan süreyi aşan görevleri sonlandırır

Süreçler

- İşletim sistemi tarafından yürütülen programlardır.
- Bir programın çalıştırılabilmesi için gerekli tüm bilgiyi tutan konteyner olarak düşünülebilir.
- Süreç tablosunda tutulurlar. İşletim sistemi tarafından atanmış kaynaklar (örneğin bellek, CPU) ile ilişkilidir.
- Diğer süreçlerle konuşabilirler (IPC).
- Bellekte saklanır ve yürütülürler. Adres uzayı ile ilişkilidir.
- Adres uzayı: 0-4G, yürütülebilir program, programın verileri ve yığını
- Yazmaçlar, dosyalar, alarmlar, ilgili süreçler…

Süreçler

Süreç ağacı. A süreci, B ve C olmak üzere 2 çocuk süreç başlatır. B süreci D, E ve F olmak üzere 3 çocuk süreç başlatır.



Adres Uzayı

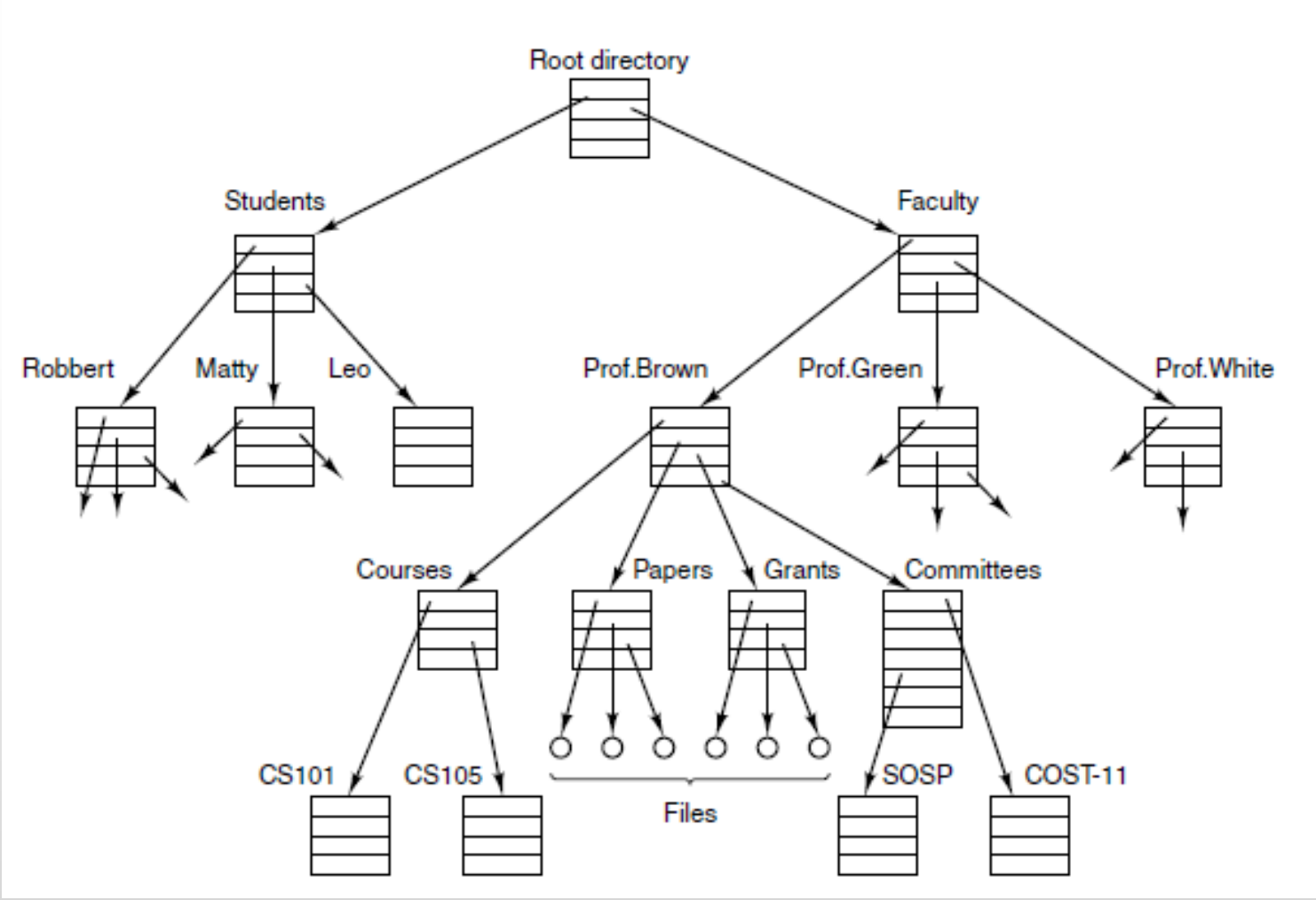
- Kavram olarak bir süreç tarafından kullanılan bellek
- İşletim sistemi, bellekte aynı anda birden çok sürece izin verir
- Bazı işlemler, fiziksel olarak mevcut olandan daha fazla belleğe ihtiyaç duyar, bu durumda sanal bellek devreye girer.

Dosyalar

- Veri depolama birimleridir.
- Dosyalar, veri, metin, resim, video, ses ve diğer türlerde olabilir.
- İşletim sistemi tarafından yönetilir ve veri depolama işlemleri gerçekleştirilir.
- İşletim sistemi tarafından belirlenen dizin yapısına göre saklanır.
- Kullanıcılar tarafından erişilebilir, okunabilir, yazılabilir veya silinebilir.
- Blok tabanlı (disk), karakter tabanlı (yazıcı, modem) olabilir.

Dosyalar

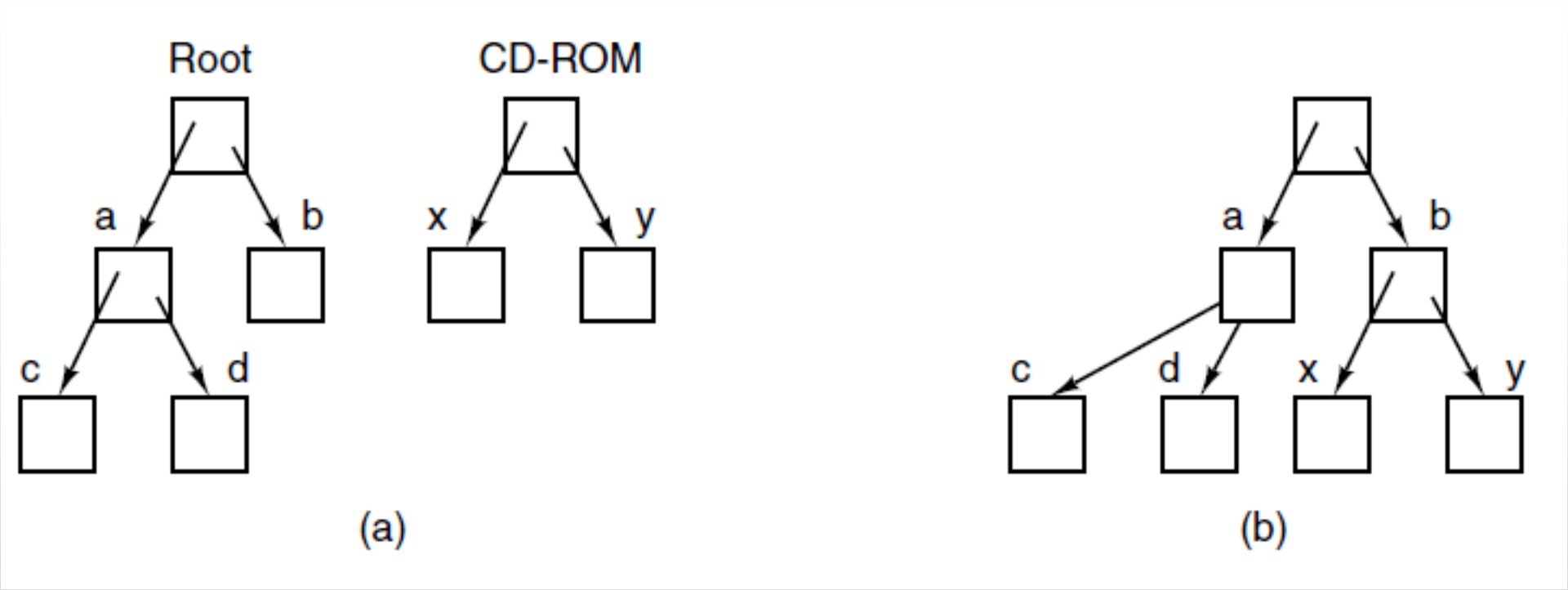
- Örnek dosya sistemi.



Dosyalar

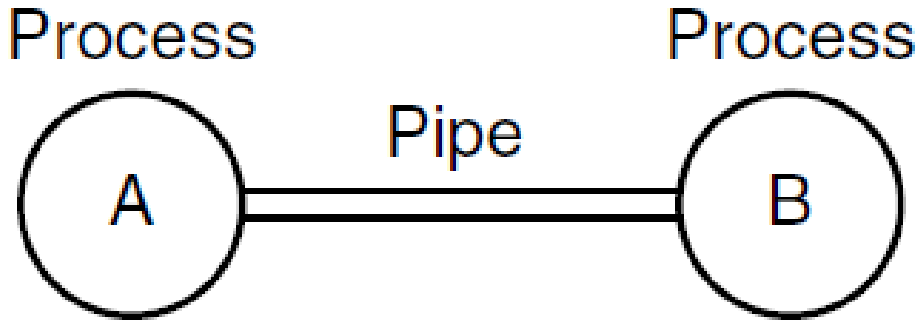
(a) Bağlamadan (mount) önce, CD-ROM'daki dosyalara erişilemez.

(b) Bağlandıktan sonra, dosya hiyerarşisinin bir parçasıdırlar.



Dosyalar

- 2 süreç boru (pipe) ile bağlanmış



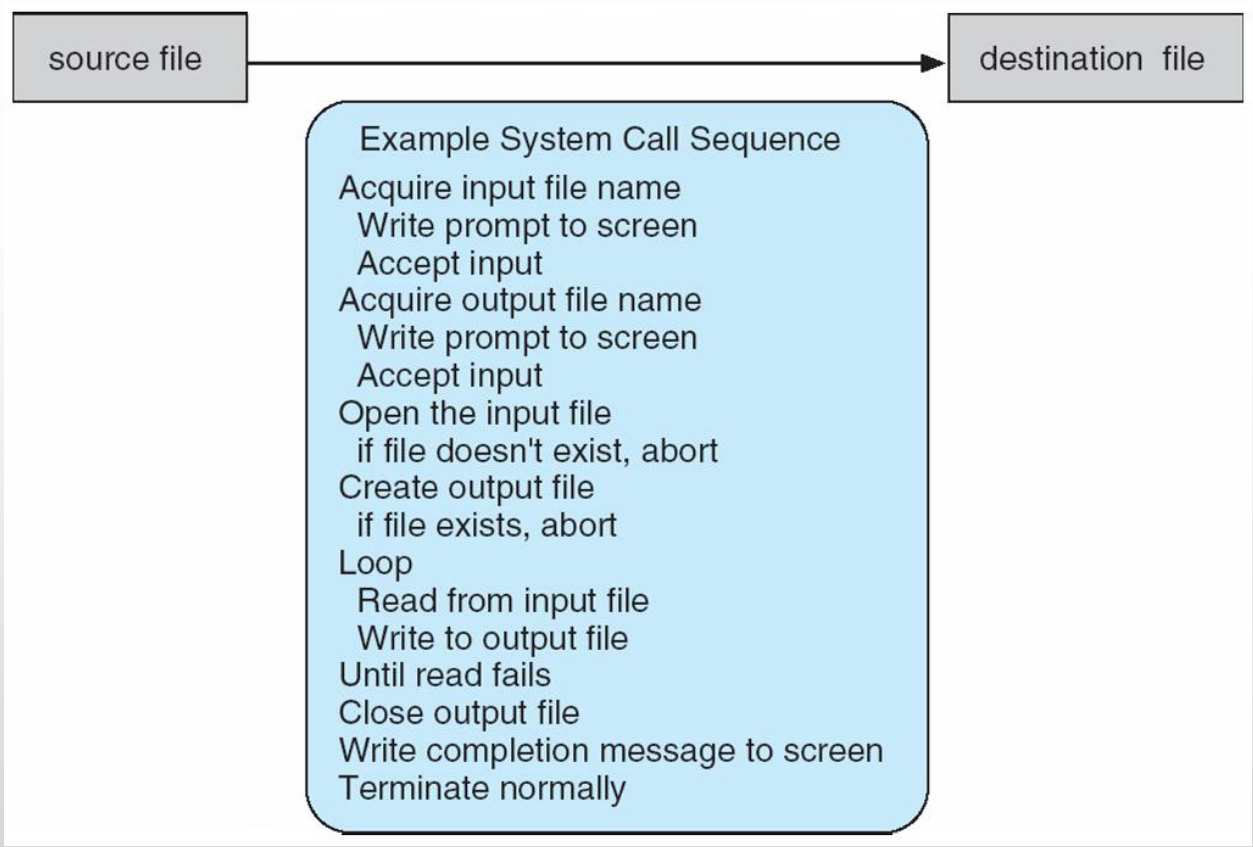
- Boru, iki süreci birbirine bağlayan ve veri gönderip alarak iletişim kurmalarını sağlayan tek yönlü bir veri kanalıdır.
- Anonim (anonymous), ebeveyn ve çocuk süreçler gibi ilgili süreçler arasında IPC için kullanılır.
- Adlandırılmış (named), iki ayrı uygulama arasındaki ilgisiz süreçler arasında IPC için kullanılır.

Sistem Çağrıları

- Sistem çağrıları, işletim sistemi tarafından sağlanan hizmetlere erişmek için kullanılır.
  - Örneğin, dosya işlemleri, bellek yönetimi, zaman hizmetleri, vb.
- İşletim sistemi tarafından tanımlanmış bir arayüze göre gerçekleştirilir.
- Sistem çağrıları, uygulama programları tarafından kullanılır ve işletim sistemi tarafından yürütülür.
- Sistem çağrıları sistemden sisteme değişir, ancak temel kavramlar benzerdir.

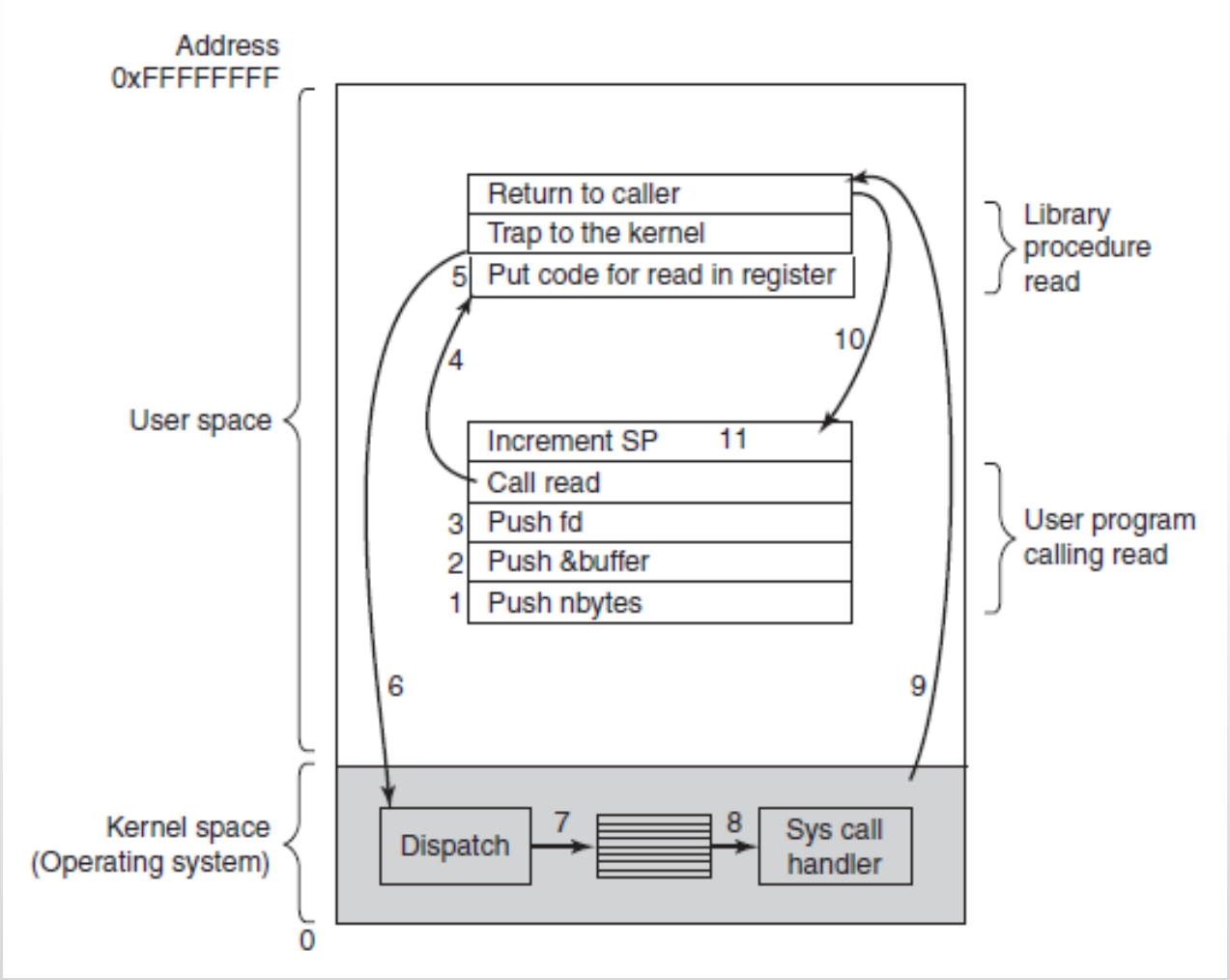
Örnek Sistem Çağrısı

- Bir dosyanın içeriğinin başka bir dosyaya aktarılması

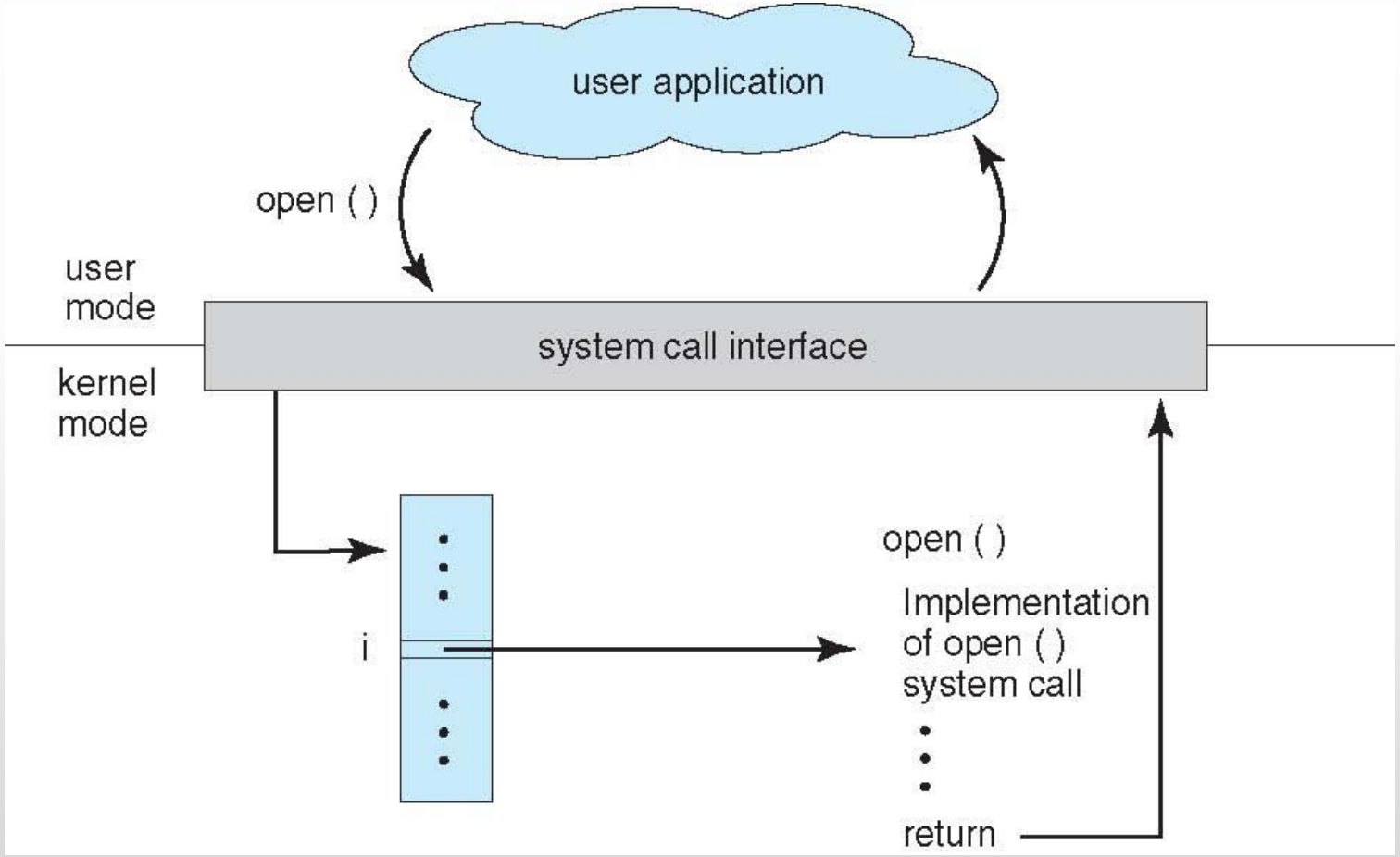


Sistem Çağrıları

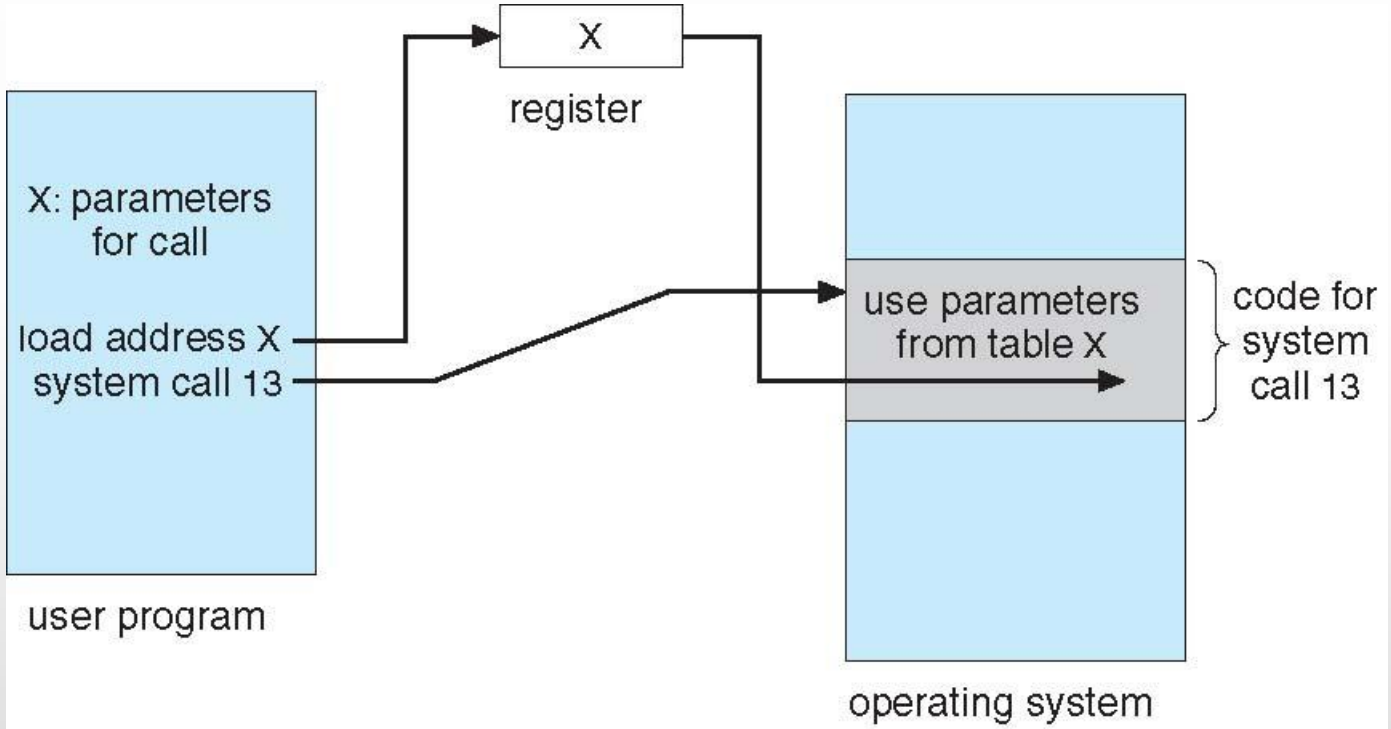
read(fd, buffer, nbytes) sistem çağrısının adım adım gösterimi



Sistem Çağrıları



Sistem Çağrıları – Parametre Aktarımı



Sistem Çağrıları – Süreç Yönetimi

Başlıca POSIX sistem çağrıları. Hata durumunda -1 döner.

pid: işlem kimliği.

s: geri dönüş kodu

Çağrı	**Tanım**
pid = fork()	Ebeveyn ile aynı bir alt süreç oluşturur
pid = waitpid(pid, &statloc, options)	Çocuk sürecin sonlanmasını bekler
s = execve(name, argv, environp)	Bir sürecin ana görüntüsünü değiştirir
exit(status)	Süreci yürütmeyi durdurur ve durumu geri döndürür

Sistem Çağrıları – Dosya Yönetimi

- fd: dosya tanıtıcı,
- n: bayt sayısı,
- position: dosya içinde göreli konum (offset).

Çağrı	Tanım
fd = open(file, how,...)	Okumak, yazmak veya her ikisi için bir dosya açar
s = close(fd)	Açık bir dosyayı kapatır
n = read(fd, buffer, nbytes)	Bir dosyadan arabelleğe veri okur
n = write(fd, buffer, nbytes)	Arabellekten bir dosyaya veri yaz
position = lseek(fd, offset, whence)	Dosya işaretçisini hareket ettirir
s = stat(name, &buf)	Bir dosyanın durum bilgisini alır

Sistem Çağrıları – Dizin Yönetimi

- .

Çağrı	Tanım
s = mkdir(name, mode)	Yeni bir dizin oluşturur
s = rmdir(name)	Boş bir dizini kaldırır
s = link(name1, name2)	name1'i işaret eden yeni bir name2 adında girdi oluşturur
s = unlink(name)	Bir dizin girdisini kaldırır
s = mount(special, name, flag)	Bir dosya sistemini bağlar
s = umount(special)	Bir dosya sisteminin bağlantısını keser

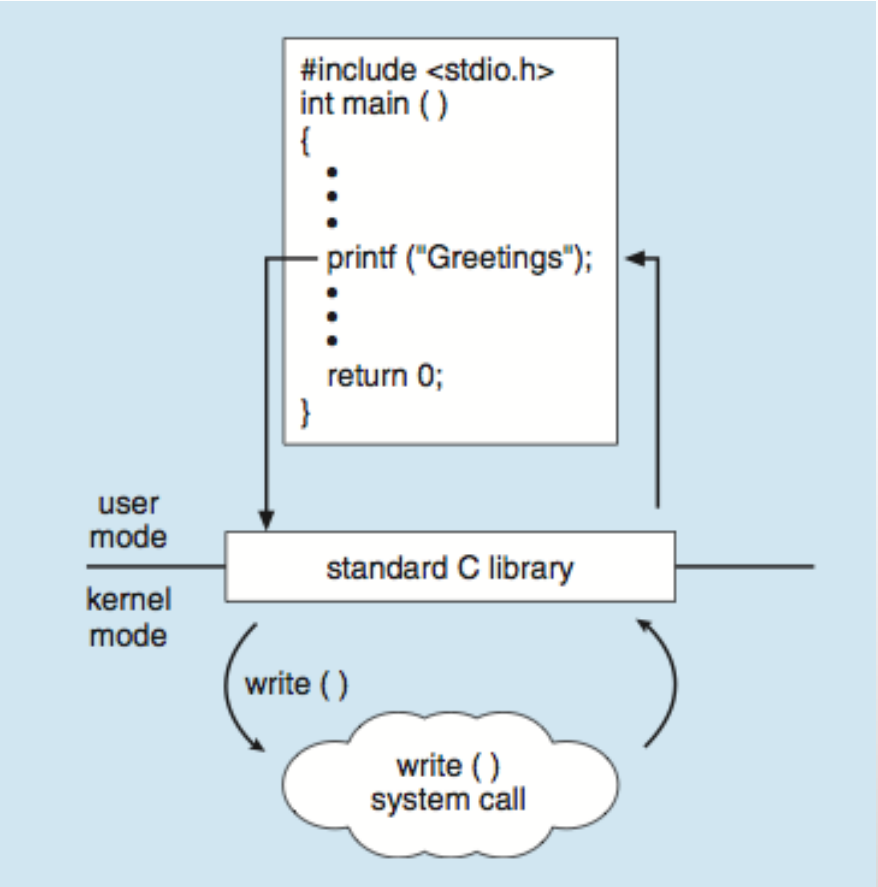
Sistem Çağrıları

seconds: geçen süre

Çağrı	Tanım
s = chdir(dirname)	Çalışma dizinini değiştirir
s = chmod(name, mode)	Bir dosyanın koruma bitlerini değiştirir
s = kill(pid, signal)	Bir sürece sinyal gönderir
seconds = time(&seconds)	1 Ocak 1970'ten bu yana geçen süreyi döndürür

Standart C Kütüphanesi

- .



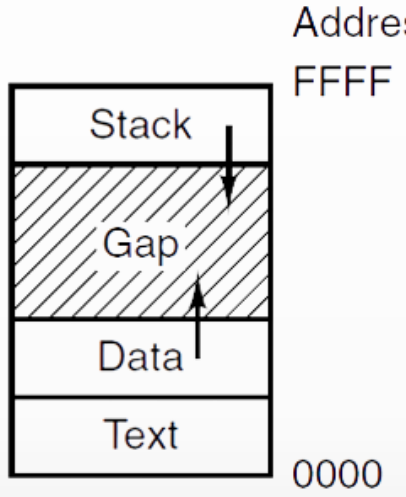
Süreç Yönetimi

```
#define TRUE 1

while (TRUE) {                                  /* repeat forever */
        type_prompt( );                         /* display prompt on the screen */
        read_command(command, parameters);      /* read input from terminal */

        if (fork( ) != 0) {                     /* fork off child process */
             /* Parent code. */
             waitpid(–1, &status, 0);           /* wait for child to exit */
        } else {
             /* Child code. */
             execve(command, parameters, 0);    /* execute command */
        }
}
```

Süreçlerin Bellek Yönetimi

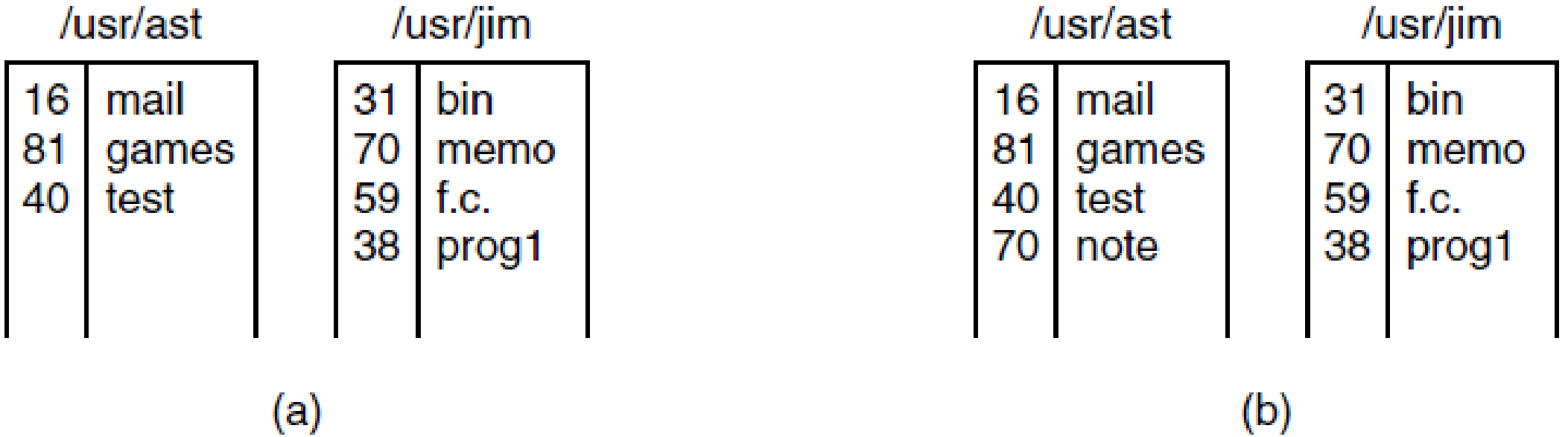


- Süreçler 3 kesime sahiptir. metin, veri, yığın
- **Yığın**, bir işlev çağrısı için gerekli olan parametreler, yerel değişkenler ve işlev dönüş adresleri gibi geçici verileri depolar.
- **Veri**, genel ve statik değişkenlerin yanı sıra dinamik olarak ayrılmış belleği depolar. Tüm iş parçacıkları arasında paylaşılır.
- **Metin**, sürecin yönergelerini uygulayan makine kodunu tutar. Salt okunurdur ve işlemin tüm iş parçacıkları arasında paylaşılır. Talimatları yürütmek için CPU tarafından kullanılır.

Dizin Yönetimi

(a) usr/jim/memo'yu ast'nin dizinine bağlamadan önce.

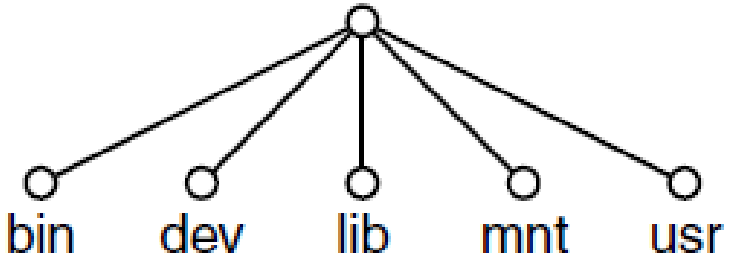
(b) bağlandıktan sonra.



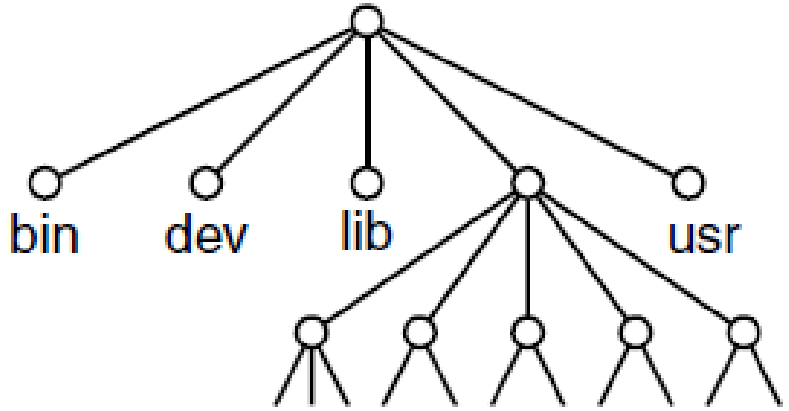
Dizin Yönetimi

(a) Bağlamadan önce dosya sistemi

(b) Bağlamadan sonra



(a)

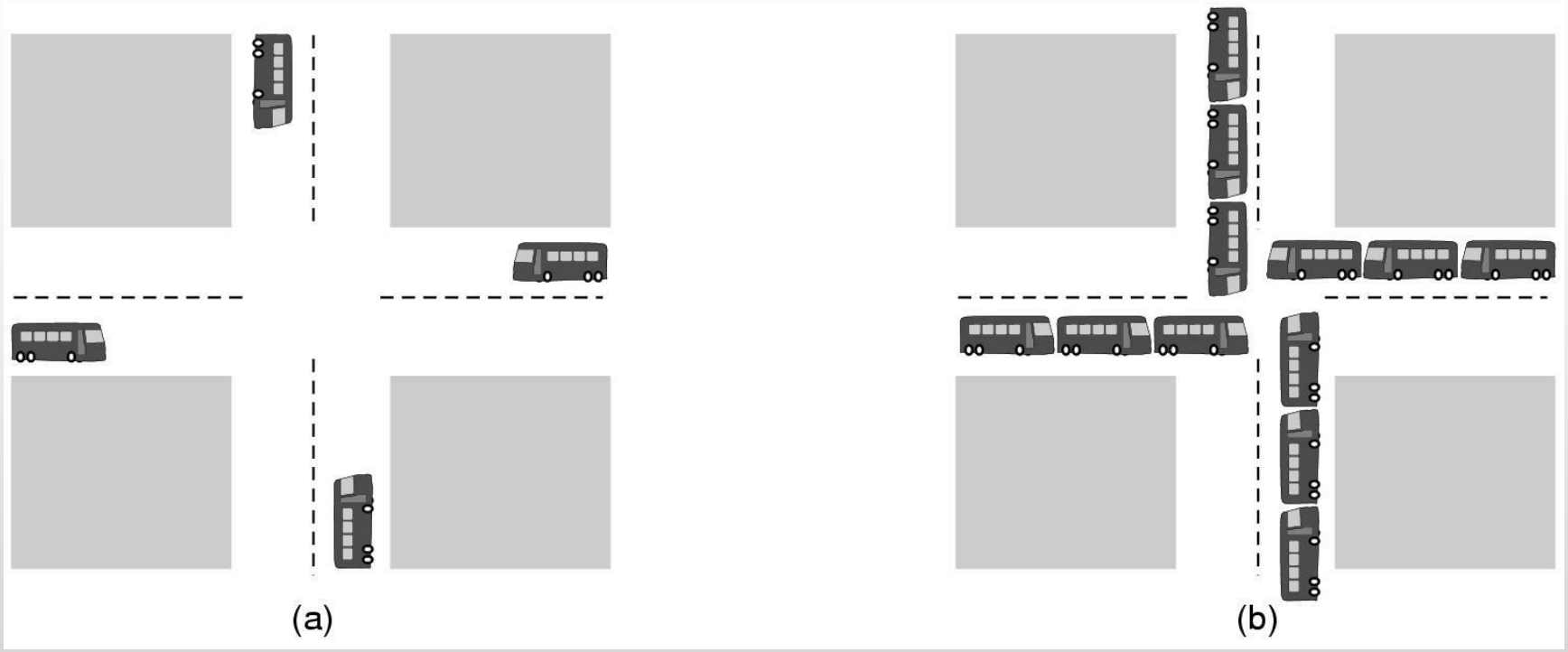


(b)

Kilitlenme (Deadlock)

(a) Potansiyel kilitlenme

(b) Gerçekleşmiş kilitlenme



The Windows Win32 API

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory

The Windows Win32 API

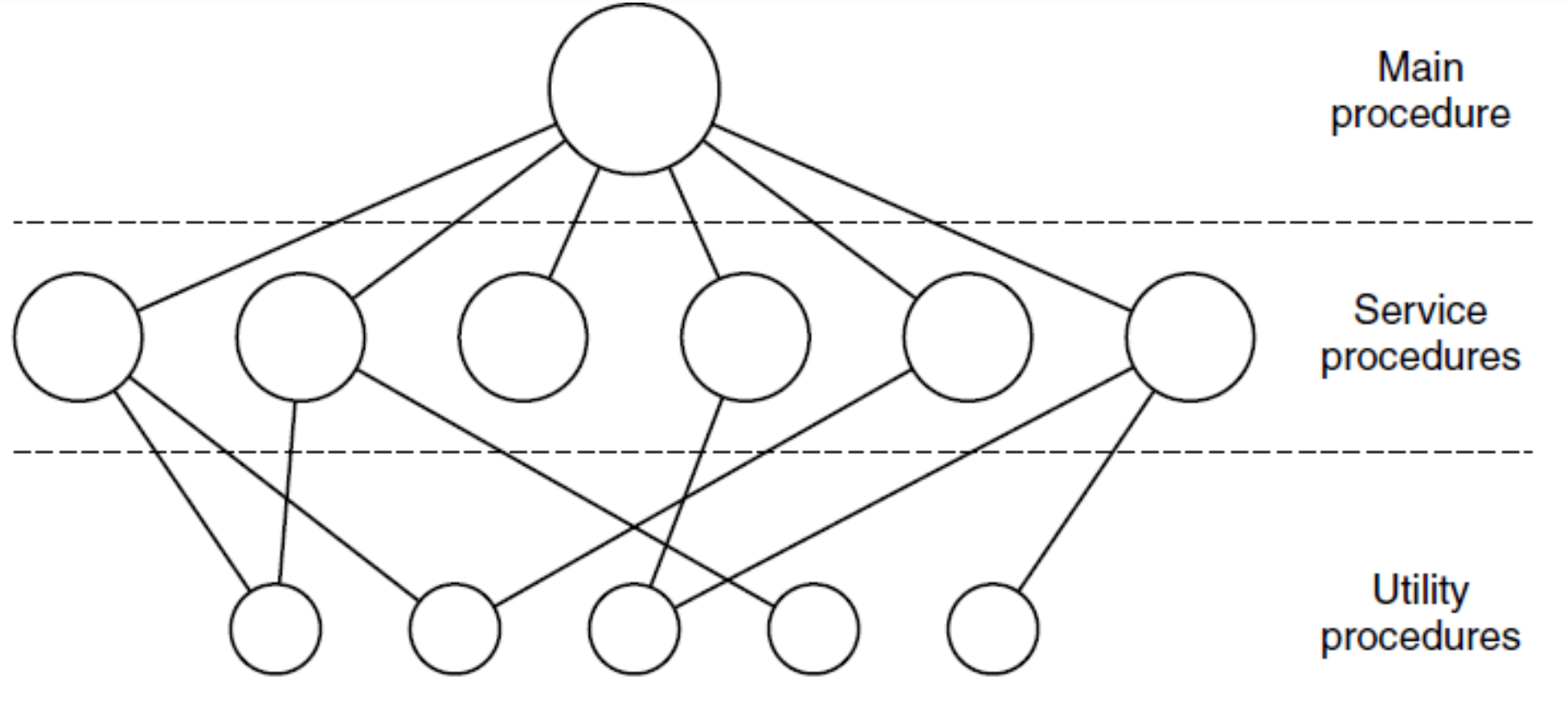
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

Monolitik Sistem

İşletim sisteminin temel yapısı

- İstenen hizmet prosedürünü başlatan bir ana program.
- Sistem çağrılarını gerçekleştiren bir dizi hizmet prosedürü.
- Hizmet prosedürlerine yardımcı olan bir dizi yardımcı prosedür.

Monolitik Sistem Yapısı



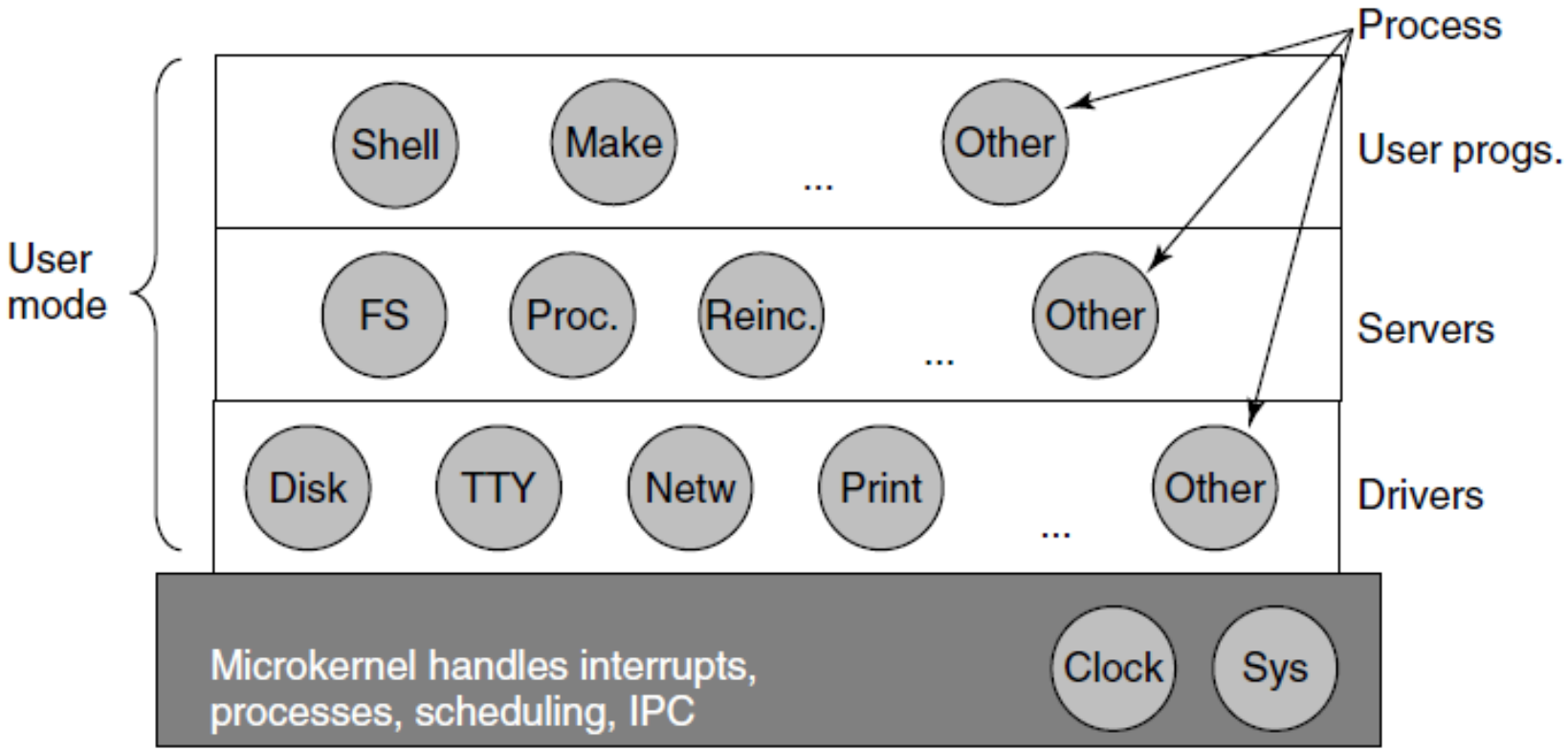
Katmanlı Sistem Yapısı

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

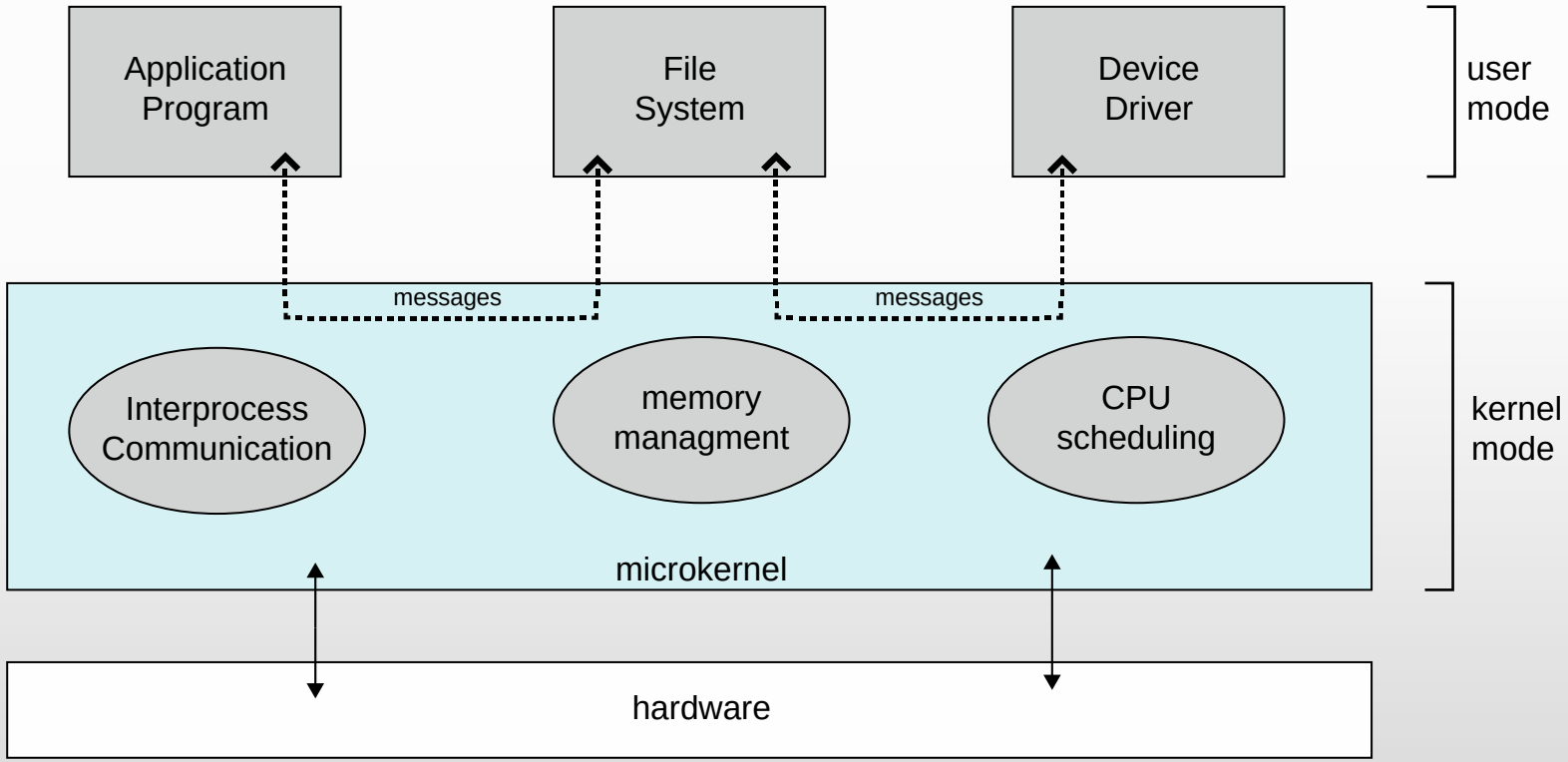
Mikro Çekirdek

- Çekirdekte az sayıda sürecin yürütülmesine izin verilir
- Hataların etkilerini en aza indirir
  - Sürücüdeki bir hatanın sistemi çökertmesi istenmez
- Mekanizma çekirdekte, ilke (policy) çekirdeğin dışındadır
  - Mekanizma, süreçler önceliklerine göre çizelgelenir (kernel)
  - İlke, süreç öncelikleri kullanıcı modunda tanımlanır (user)

Mikro Çekirdek Sistem Yapısı



Mikro Çekirdek Sistem Yapısı



Andrew Tanenbaum (Monolitik çekirdek)

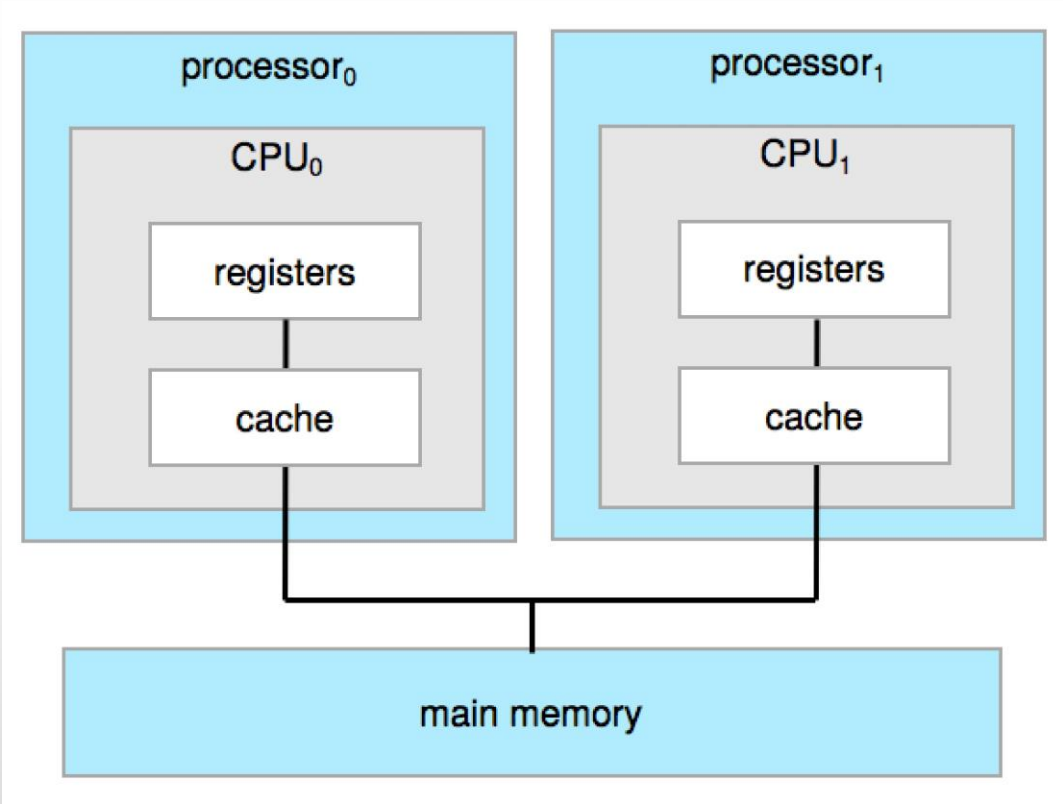
- Monolitik çekirdeklerin tasarımı ve uygulanması daha basittir ve daha bütünleşik bir sistem sağlar.
- Monolitik çekirdeklerdeki doğrudan işlev çağrıları, süreçler arası iletişim gerektiren mikro çekirdeklere kıyasla daha iyi performans sağlar.
- Çekirdekteki hatalar tüm sistemi çökertebileceğinden yekpare çekirdekler daha güvenilirdir, ancak her şeyin tek bir modülde olması hataları bulmayı ve düzeltmeyi kolaylaştırır.

Linus Torvalds (Mikro Çekirdek)

- Mikro çekirdekler daha modüler, esnek ve ölçeklenebilirdir ve daha kolay korunabilir ve geliştirilebilir.
- Mikro çekirdekler, herhangi bir bileşendeki bir hatanın verebileceği hasarı sınırlayarak daha kararlı ve güvenli bir sisteme yol açar.
- Modern bilgisayar mimarileri, mikro çekirdeklerin performans sorunlarının üstesinden gelebilir ve mikro çekirdekler, uygun şekilde tasarlandıkları takdirde daha iyi performans sunabilir.
- Linux'un açık kaynak geliştirme modeli, hızla gelişmesine ve yaygın olarak kullanılmasına yardımcı oldu.

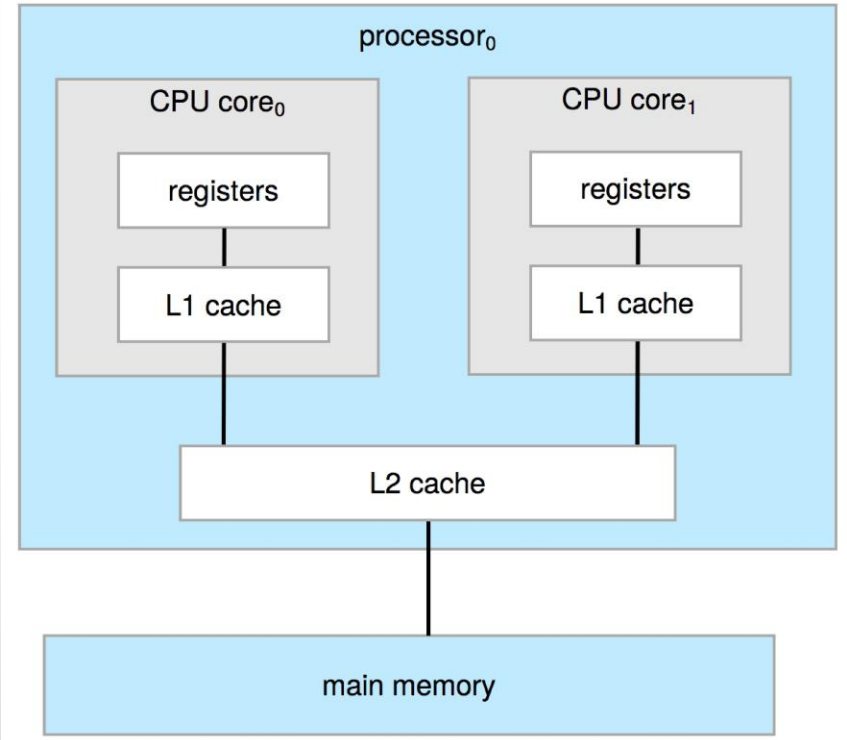
Simetrik Çoklu İşlemci Mimarisi

- .

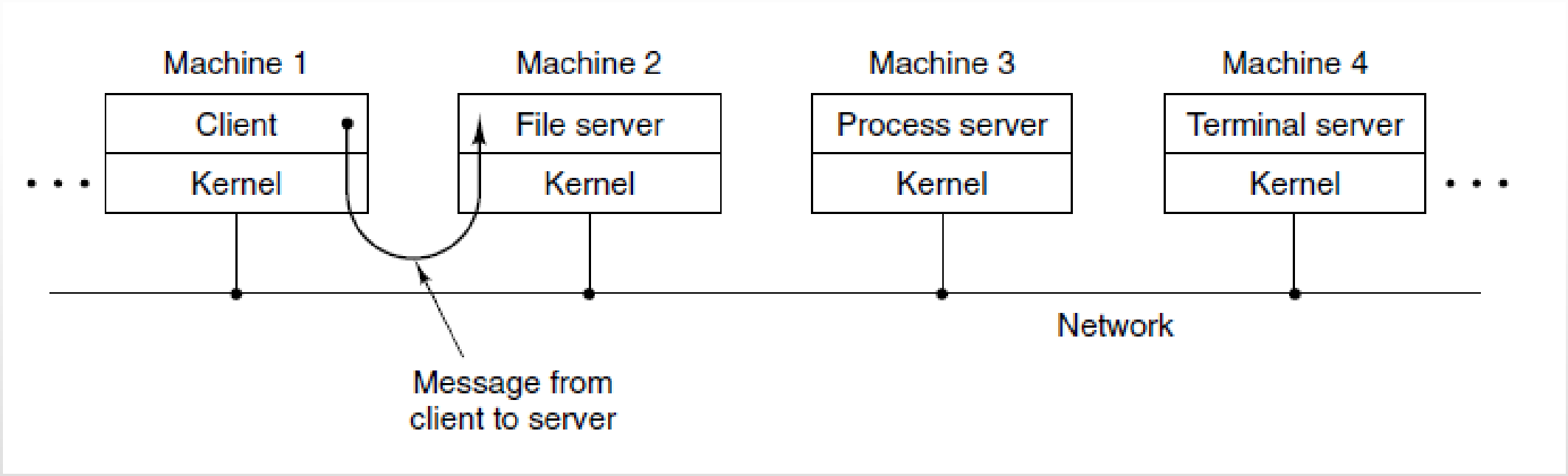


Çift Çekirdekli İşlemci Mimarisi

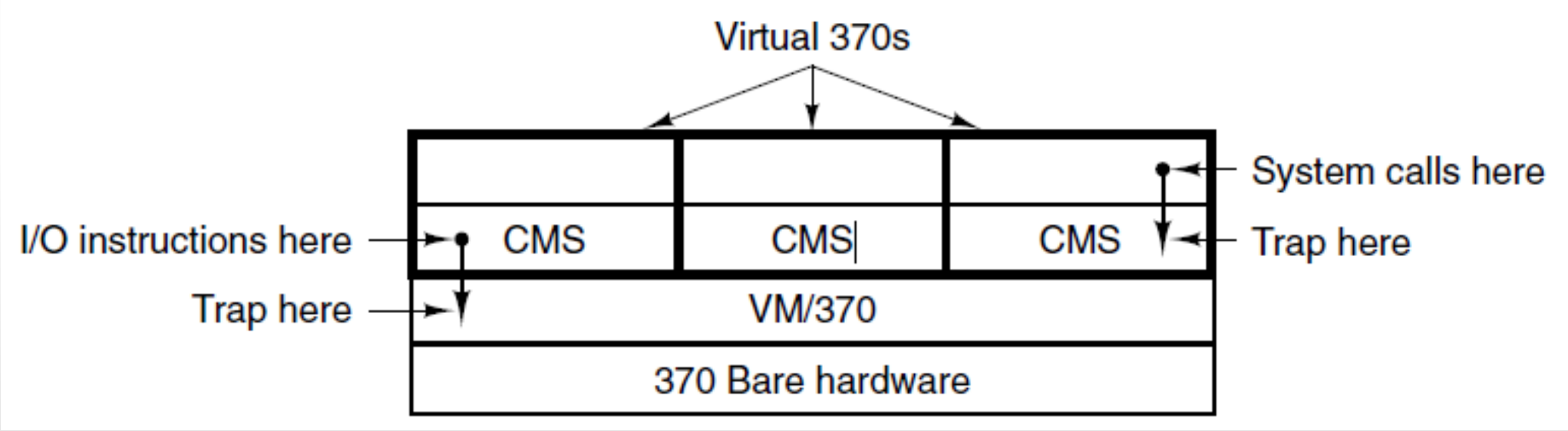
- .



İstemci Sunucu Modeli

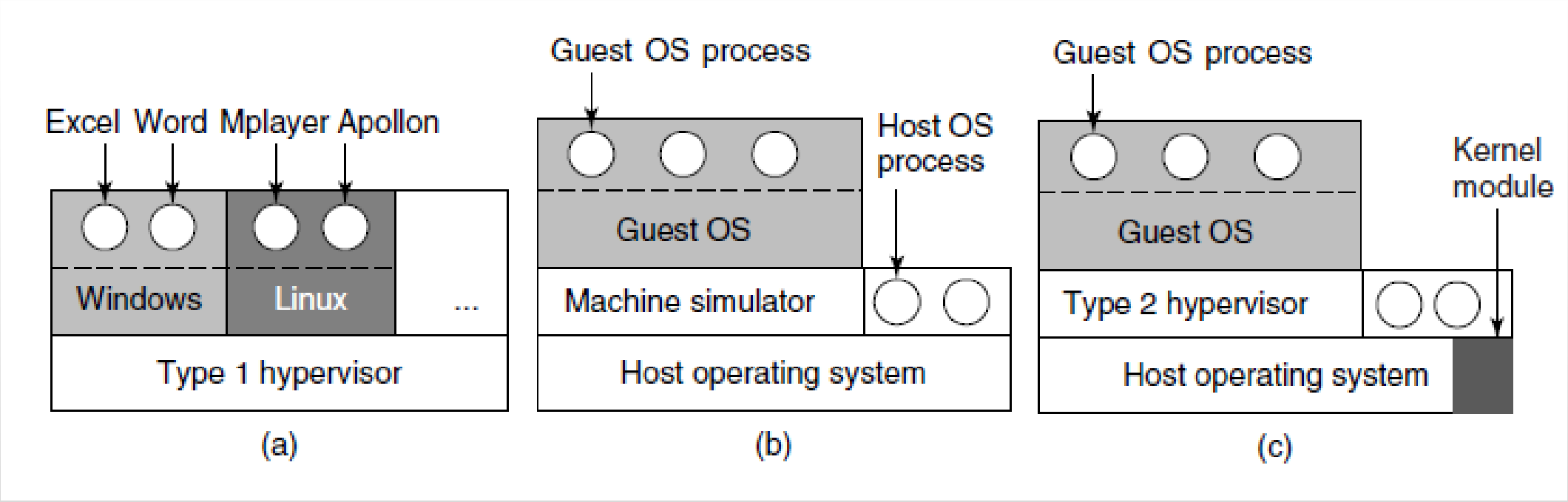


Sanal Makine Yapısı

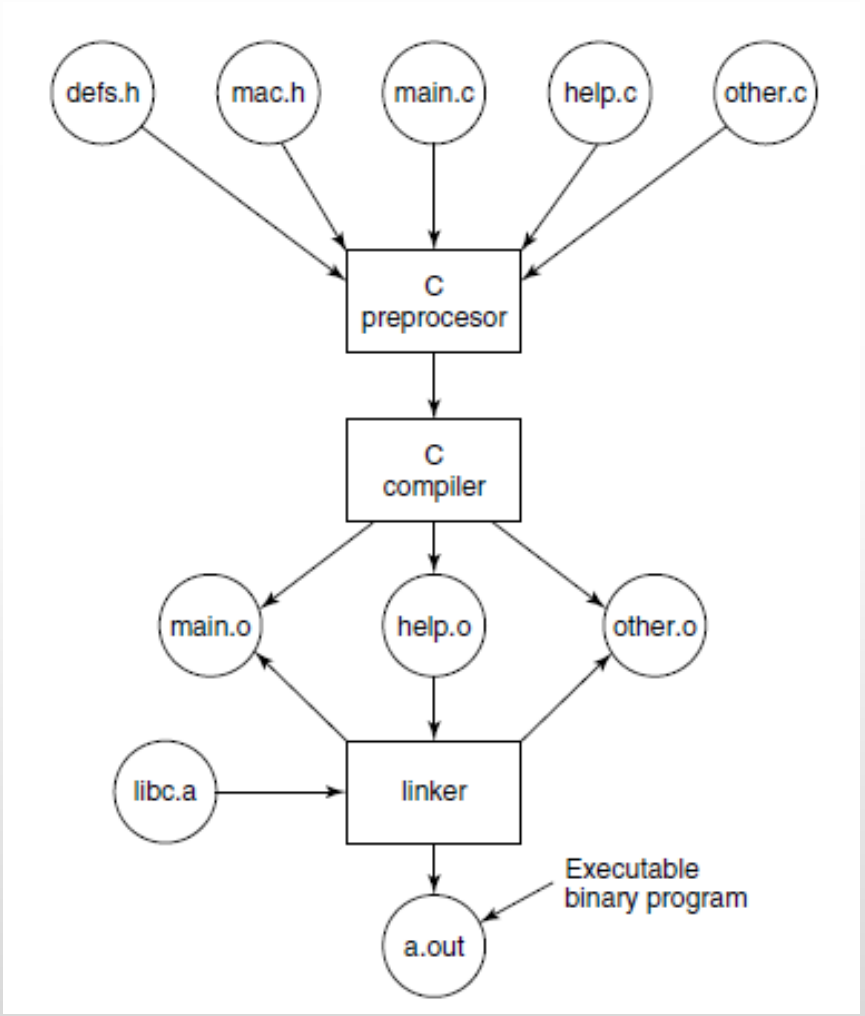


Sanal Makine Yapısı

(a) Tip 1 hipervizör. (b) Yalın tip 2 hipervizör. (c) Pratik tip 2 hipervizör.



Yürütülebilir Dosya Oluşturma



Yürütülebilir Dosya Oluşturma

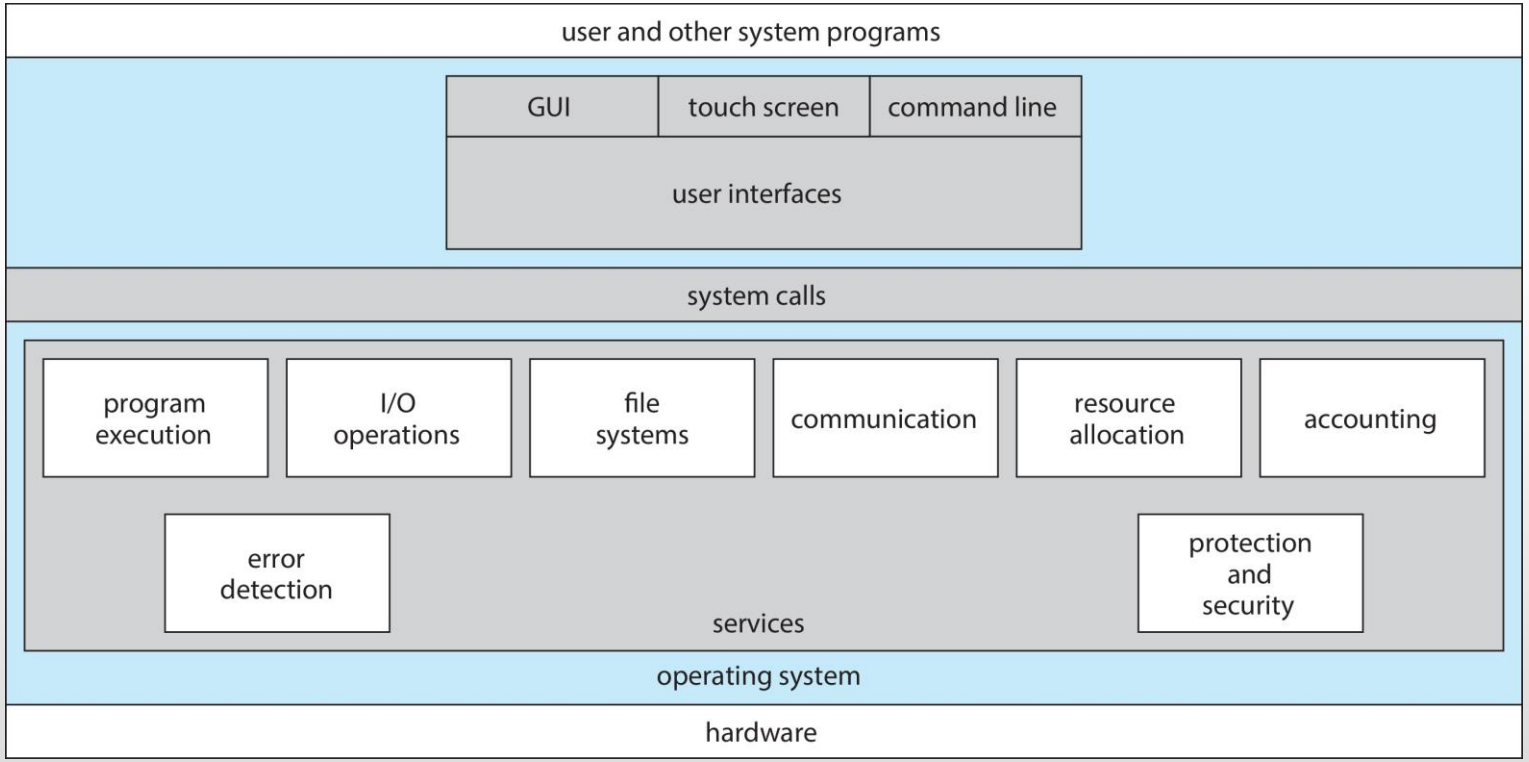
- C preprocessor (önişlemci):
  - Başlığı alır, makroları genişletir, koşullu derlemeyi ele alır.
- Compiler (derleyici)
  - .c -.o , kaynak koda göre nesne dosyalarını oluşturur.
- Linker (bağlayıcı)
  - .o uzantılı nesne dosyalarını birleştirerek yürütülebilir dosyayı oluşturur.

Metrik ve Birimleri

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^{3}	1,000	Kilo
10^{-6}	0.000001	micro	10^{6}	1,000,000	Mega
10^{-9}	0.000000001	nano	10^{9}	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

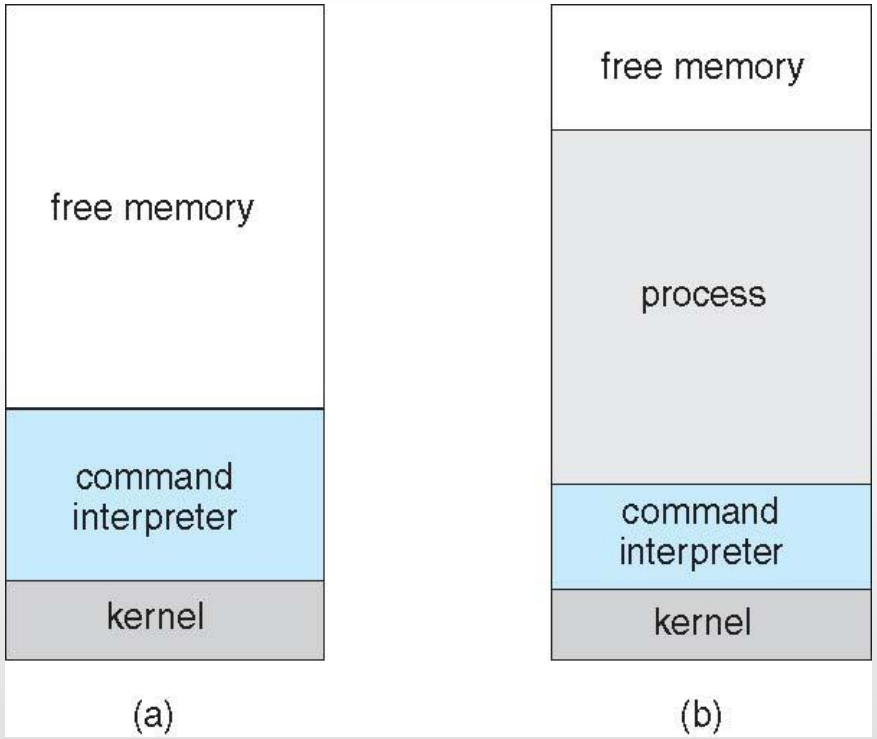
İşletim Sistemi Servisleri

- .



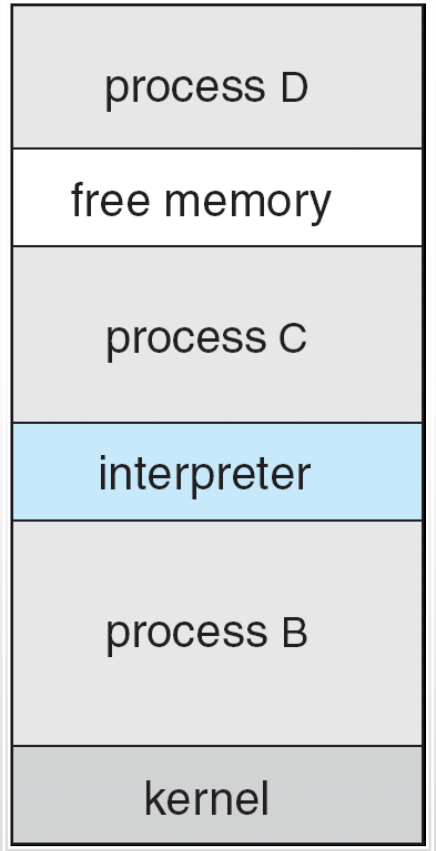
MS-DOS

- Tek görevli
- Sistem ayağa kalktığında kabuk çağrılır
- Süreç yaratılmaz
- Tek bellek alanı
- Program belleğe üzerine yazılarak yüklenir
- Program sonlandığında kabuk yeniden yüklenir
- (a) Sistem ayağa kalkarken
- (b) Program çalışırken



FreeBSD

- UNIX türevi
- Çok görevli
- Kullanıcı giriş yapar ve seçilen kabuk yüklenir
- Kabuk fork() komutu ile süreç yaratır
- exec() komutu ile program süreç içerisine yüklenir
- Kabuk programın sonlanmasını bekler veya kullanıcı komutları ile çalışmaya devam eder.



Politika ve Mekanizma

- Politika (policy): Ne yapılacak?
- Mekanizma (mechanism): Nasıl yapılır?
- Politikalar ne yapılacağını, mekanizma ise nasıl yapılacağını belirler.
- Politikanın mekanizmadan ayrılması çok önemli bir ilkedir.
- Bu ilke politika kararları daha sonra değiştirilecekse maksimum esneklik sağlar (örnek – zamanlayıcı)
- İşletim sistemi tasarlamak, oldukça yaratıcılık gerektirir.

SON