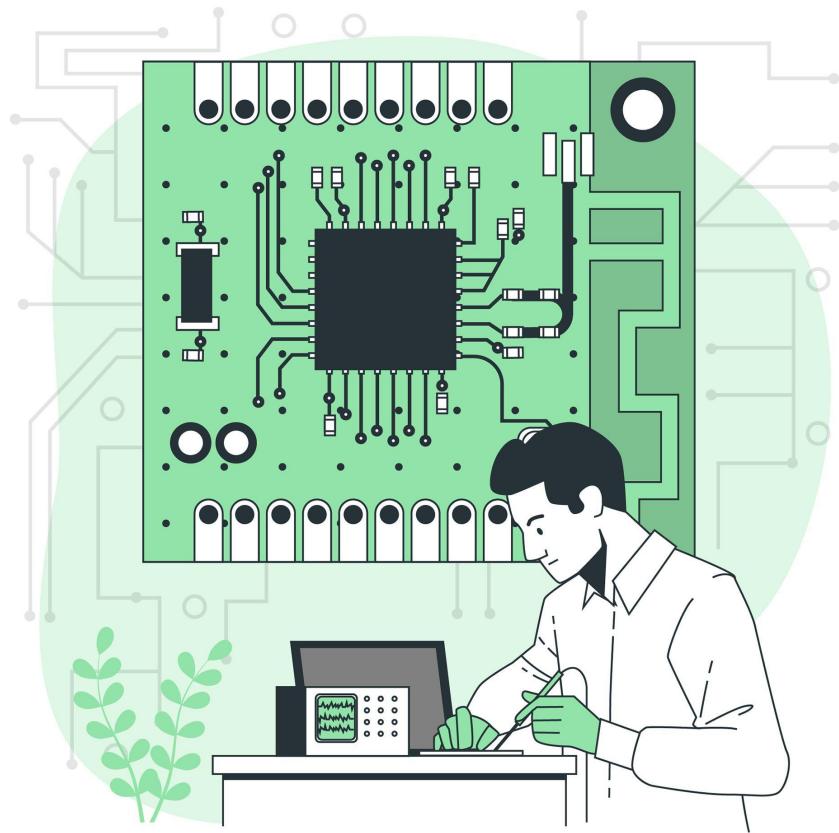


01

Giriş



Giriş

- İşletim Sistemi Nedir?
- İşletim Sistemlerinin Tarihçesi
- Bilgisayar Donanımı Genel Bakış
- İşletim Sistemi Mimarisi
- Sistem Çağrıları
- Üç Başlıkta İşletim Sistemleri



Giriş

Bilgisayar Sistemleri



İşletim Sistemi



Giriş

Most Used Mobile OS Timelapse (2009-2023)

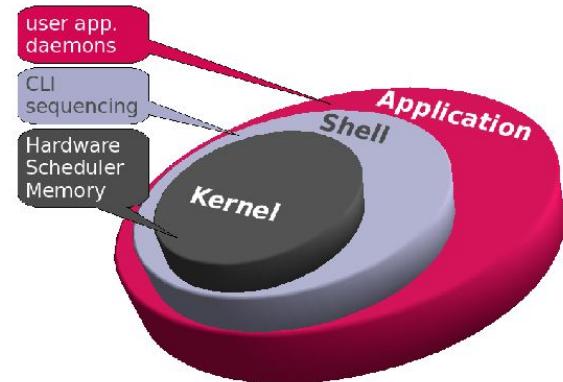
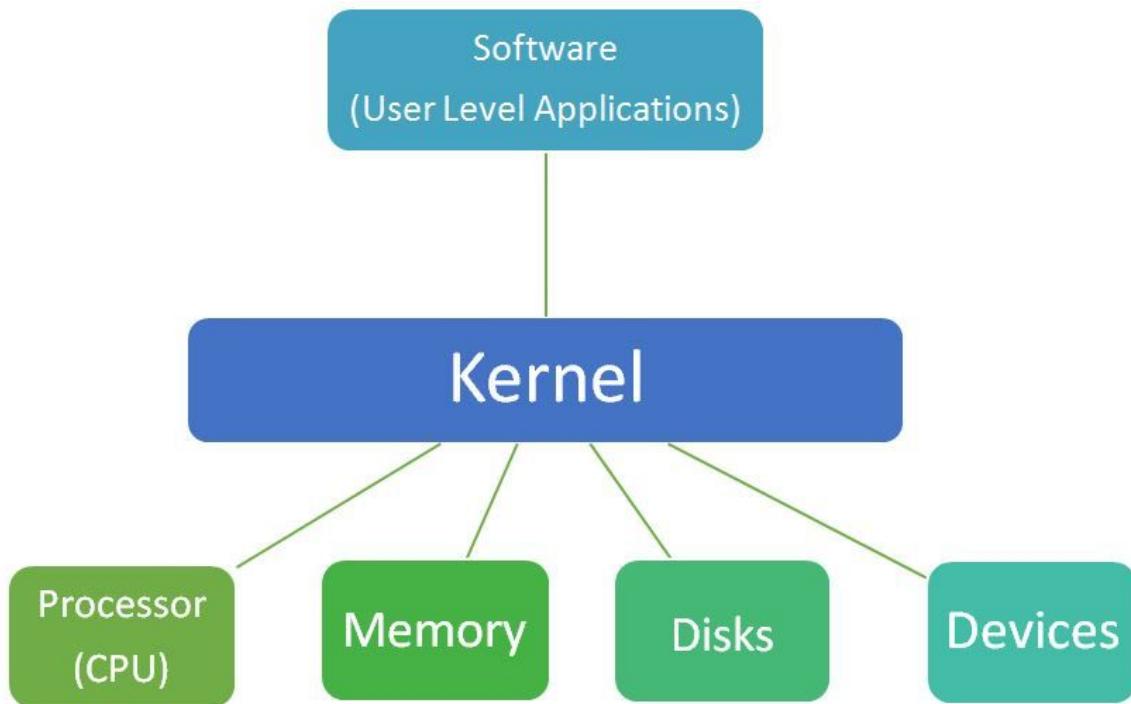


Giriş

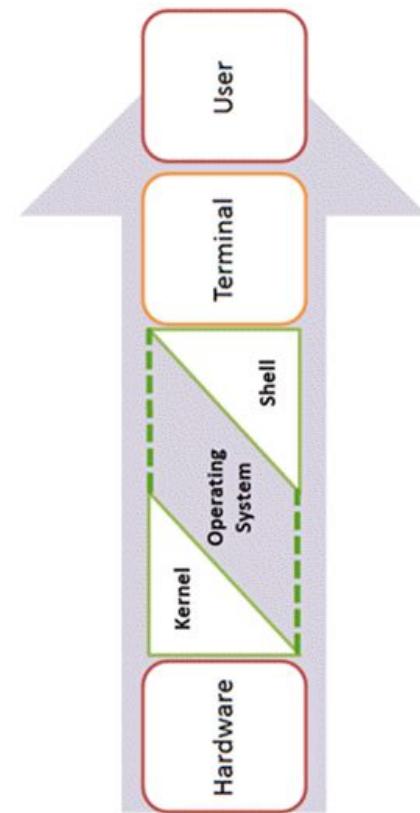
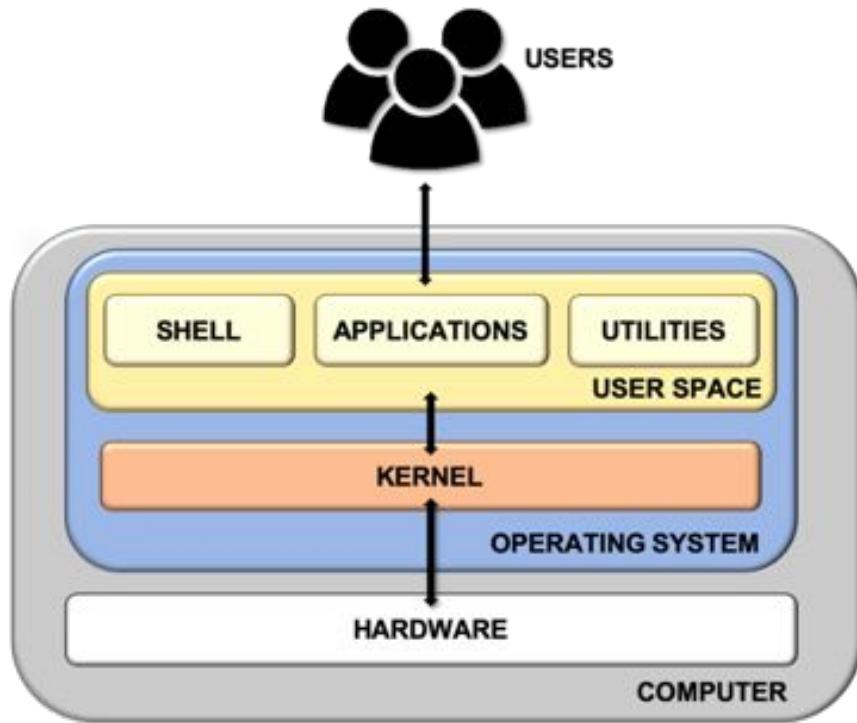


UNIX

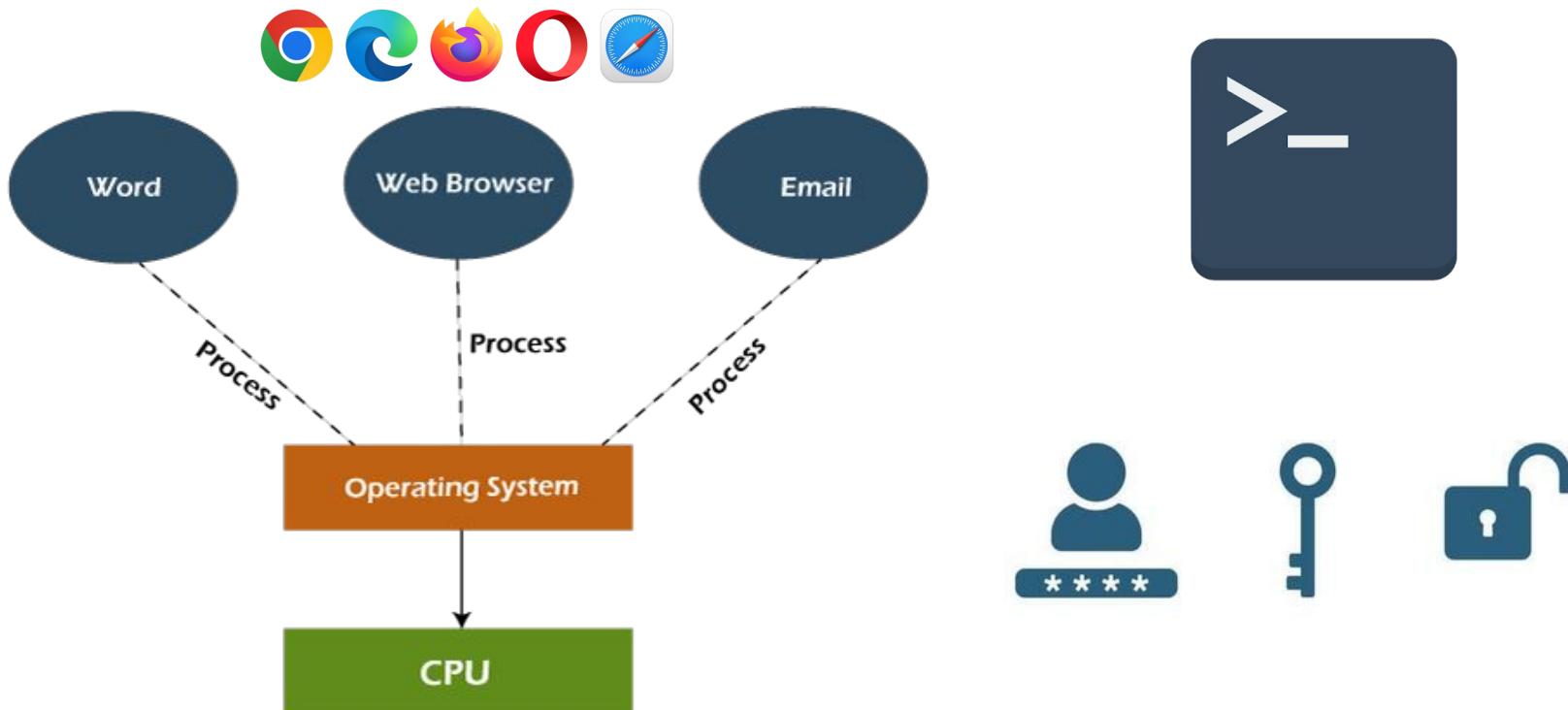
Giriş



Giriş



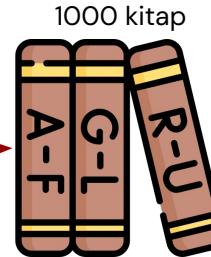
Giriş



LINES OF CODES USED



Giriş



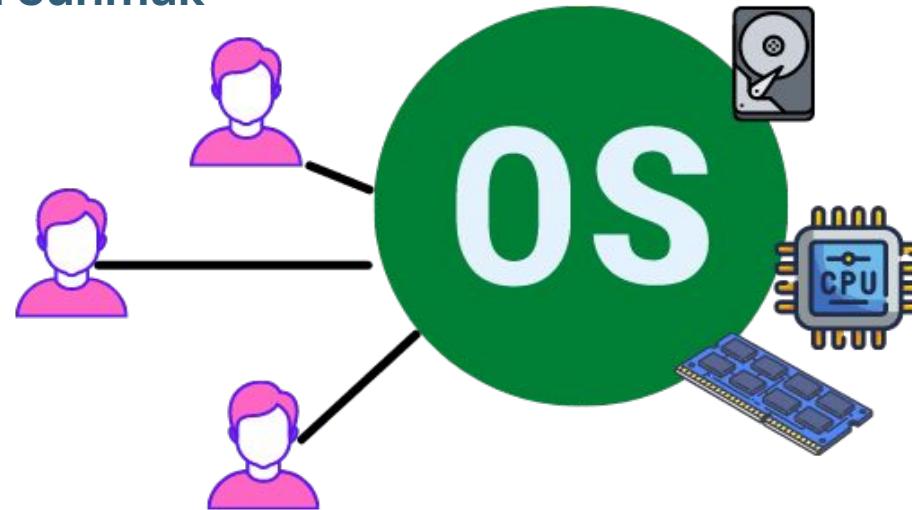
02

İşletim Sistemi Nedir?



İşletim Sistemi Nedir?

- Donanım Kaynaklarını Yönetmek
- Donanım Kaynaklarını Sunmak

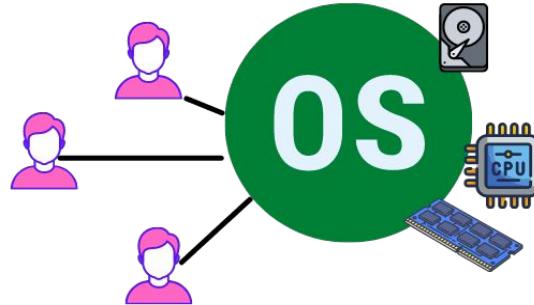


Donanım Yığını olarak İşletim Sistemi



Donanım Yığını olarak İşletim Sistemi

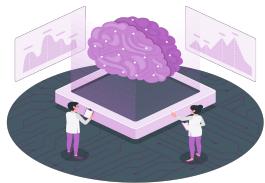
```
suhap@suhap-thinkpad-x1-yoga-gen-7:~$ googler www.kocaeli.edu.tr
1. Kocaeli Üniversitesi
https://www.kocaeli.edu.tr/
Universitemizden ÖNEMLİ GELİŞMELER - 2023-2024 Eğitim - Öğretim Yılı Üniversitemiz Akademik Takvimi - İngilizce
Veterlik Sınavı (YGS-Güz) - Kocaeli üniversitesi ...
2. Kocaeli Üniversitesi
https://www0.kocaeli.edu.tr/
3. Kocaeli Üniversitesi
https://www.turkiye.gov.tr/kocaeli-universitesi
4. Kocaeli Üniversitesi | zzzit
https://www.facebook.com/kou9zofficial/?locale=id_ID
5. Kocaeli Üniversitesi | Izmit
https://www.facebook.com/kou9zofficial/?locale=tr_TR
6. KOSTÜ | Kocaeli Sağlık ve Teknoloji Üniversitesi - Geleceği ...
https://kocaelisaglik.edu.tr/
7. kou9zofficial - Kocaeli Üniversitesi
https://www.instagram.com/kou9zofficial/
googler (? for help)
```



Kaynak Yöneticisi olarak İşletim Sistemi



İşlemci



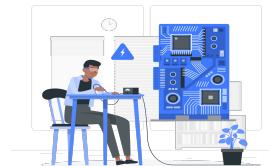
Ekran



Disk



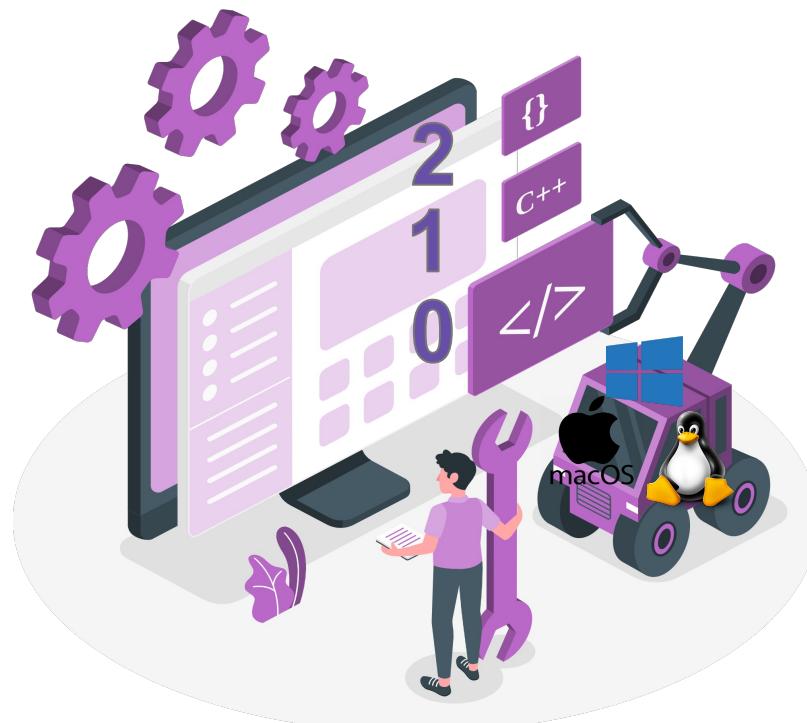
Ana Kart



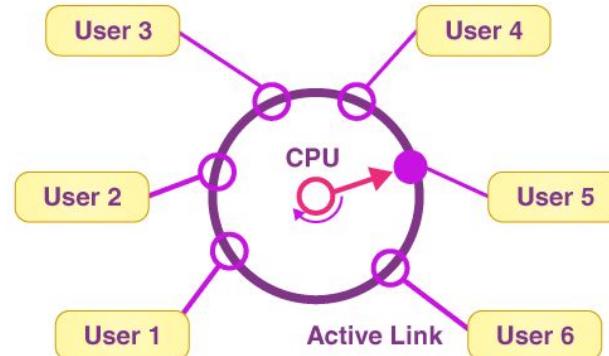
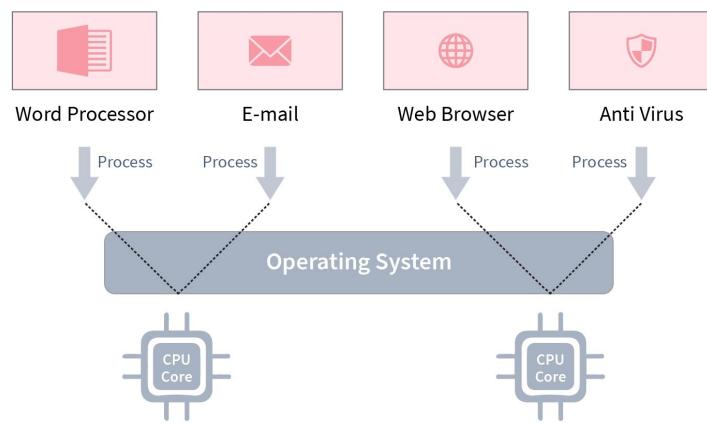
YÖNETİM



Kaynak Yöneticisi olarak İşletim Sistemi

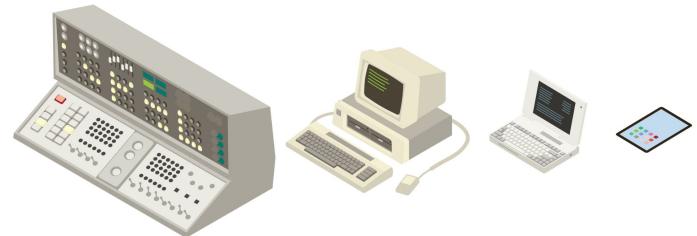


Kaynak Yöneticisi olarak İşletim Sistemi

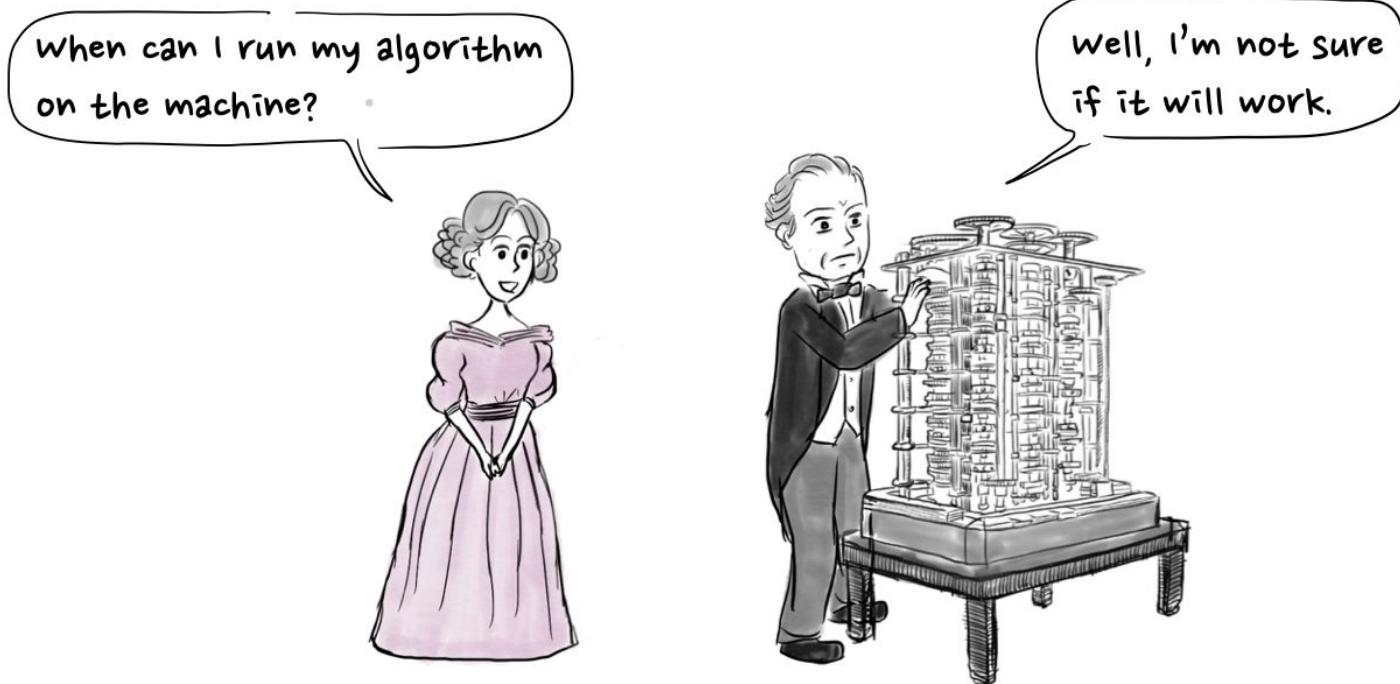


03

İşletim Sistemi Tarihçe

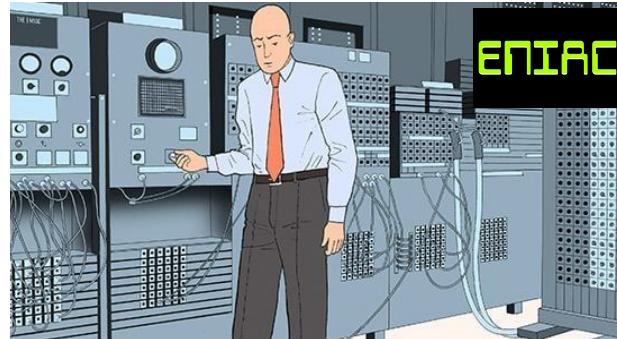
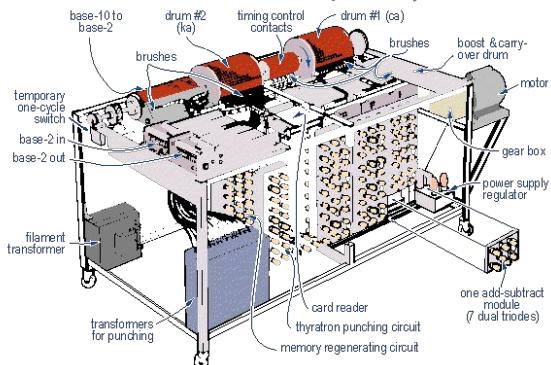


Tarihçe

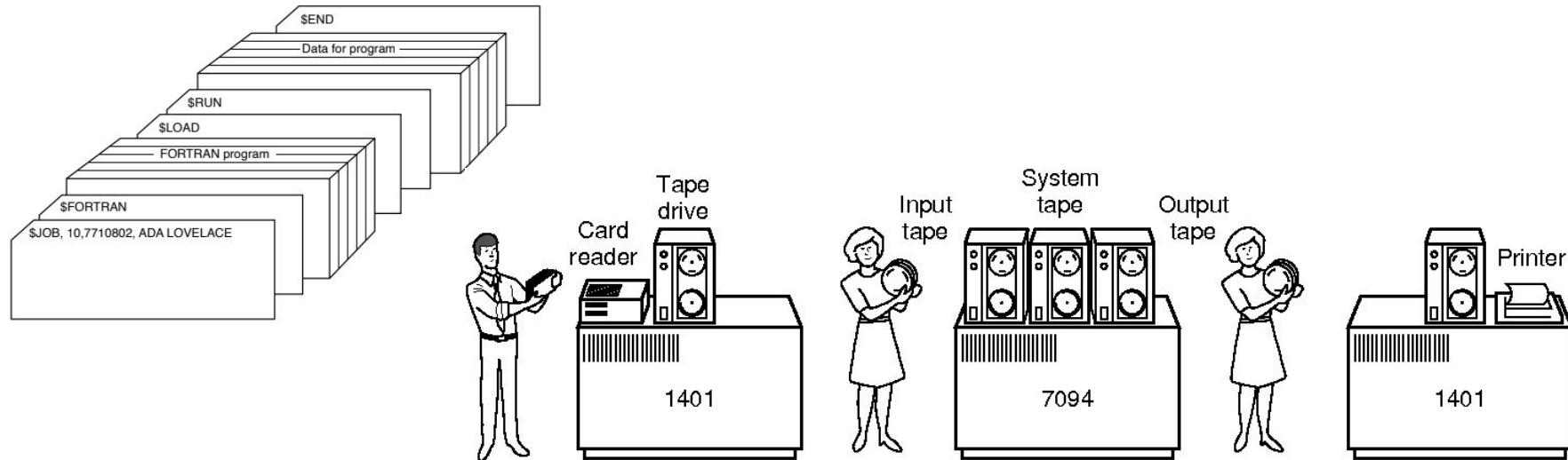


Birinci Nesil (1945-1955): Vakum Tüpler

The Atanasoff-Berry Computer



İkinci Nesil (1955-1965): Transistörler ve Toplu Sistemler



Üçüncü Nesil (1965-1980): Tümlüşik Devreler ve Çoklu Programlama

IBM 1401 Computer System



IBM, 370, 4300, 3080 ve 3090



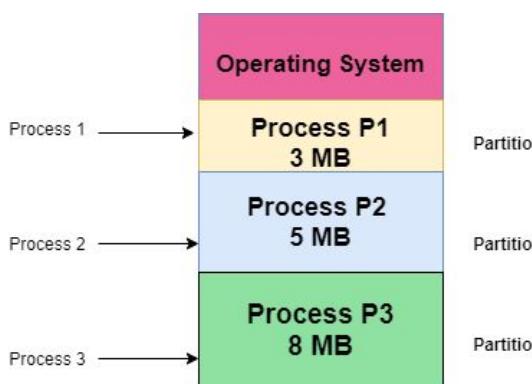
IBM 7094



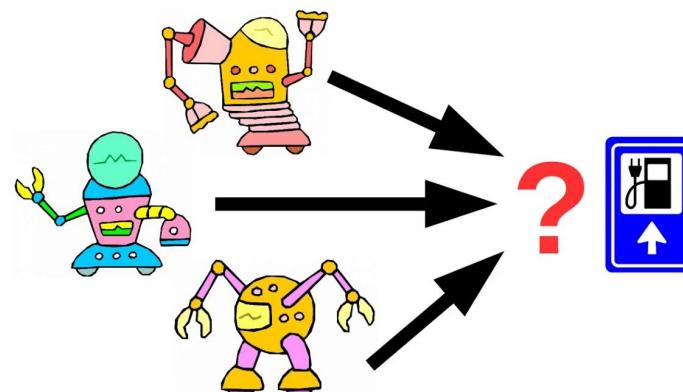
IC'leri (Integrated Circuits)



Üçüncü Nesil (1965-1980): Tümlüşik Devreler ve Çoklu Programlama



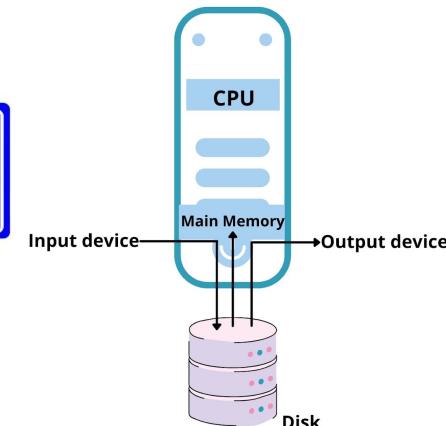
Size of Partition = Size of Process



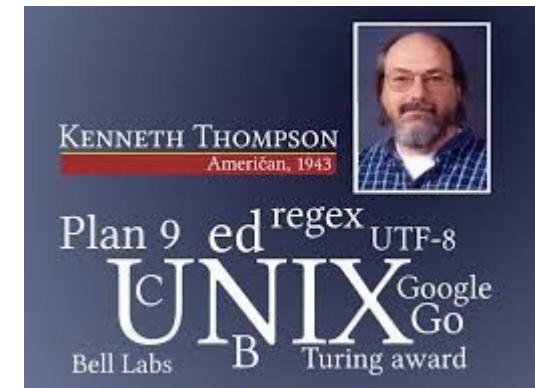
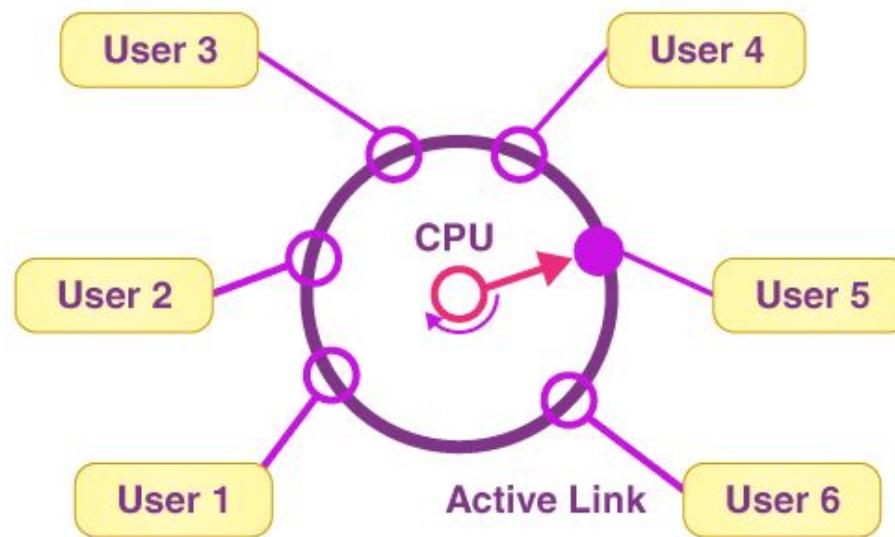
© Authors of ICRA 2018 Paper 24

Wed AM

Pod S.1



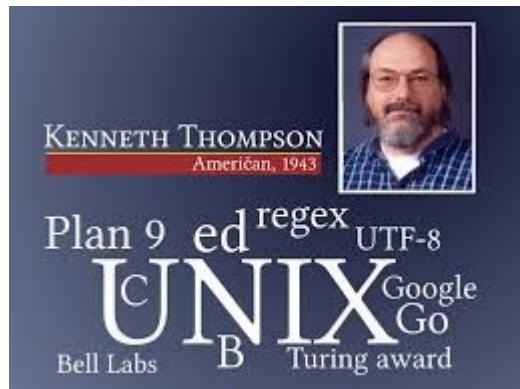
Üçüncü Nesil (1965-1980): Tümlüşik Devreler ve Çoklu Programlama



Üçüncü Nesil (1965-1980): Tümlüşik Devreler ve Çoklu Programlama



MINIX 3



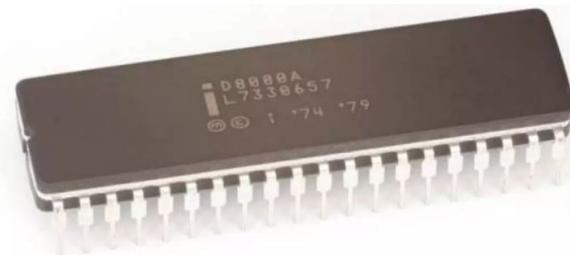
Dördüncü Nesil (1980-Günümüz): Kişisel Bilgisayarlar



Dördüncü Nesil (1980-Günümüz): Kişisel Bilgisayarlar



 DIGITAL
RESEARCH

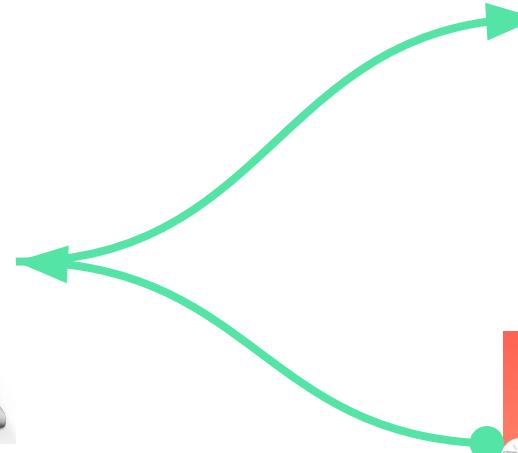


 8080 processor

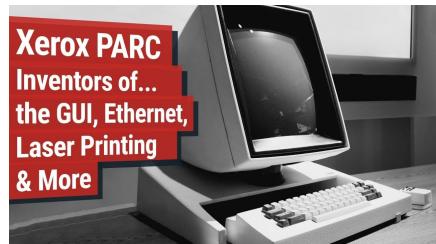
Dördüncü Nesil (1980-Günümüz): Kişisel Bilgisayarlar



GATES, BILL



Dördüncü Nesil (1980-Günümüz): Kişisel Bilgisayarlar



Dördüncü Nesil (1980-Günümüz): Kişisel Bilgisayarlar



1985 - 2001



1990 - 2001



1992 - 2001



1993 - 2001



1994 - 2001



GATES, BILL



1995 - 2001



1996 - 2004



1998 - 2006



2000 - 2006



2000 - 2010



2001 - 2014



2006 - 2017



2009 - 2020



2012 - 2016



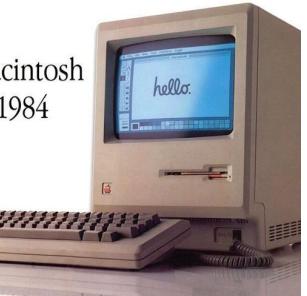
2013 - now



2015 - now

2020 - now

2021 - now



Introducing Macintosh
January 24, 1984

Beşinci Nesil (1990-Günümüz): Mobil Bilgisayarlar

1938

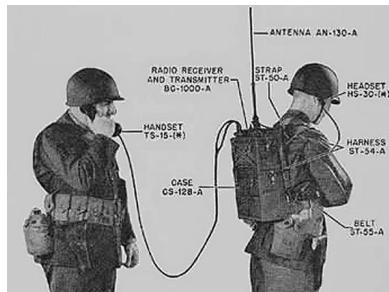
SCR-194 ve 195



ABD
Taşınabilir telsiz
11 Kg
8 Km

1940

SCR-300



Motorola
Taşınabilir radyo
17 Kg
5 Km

1942

SCR-536



Motorola
Handie Talkie
2 Kg
1,5 Km

Beşinci Nesil (1990-Günümüz): Mobil Bilgisayarlar

1946

Mobil Telefon Sistemi (MTS)



Bell
80 Kg
5 Km
Arama başına ayda 330 \$

1956

Mobil Sistem A (MTA)



Ericsson
40 Kg
5 Km

1964

Mobil Telefon Hizmetinin (IMTS)



Motorola
Mobil Telefon Hizmeti
18 Kg
Yüksek Fiyat
Kısıtlı kullanım

1982

DynaTAC
(Dinamik Uyarlanabilir Toplam Alan Kapsamı)



Motorola
10 Kg
1G
60 dak. konuşma

Beşinci Nesil (1990-Günümüz): Mobil Bilgisayarlar

1983
DynaTAC



Motorola
1 Kg
9000\$

1984
Mobira Talkman



Mobira
Uzun Konuşma Süresi

1992
International 3200



Motorola
2G

1993
IBM Simon



IBM
Akıllı telefon
-> Çağrı cihazı,
-> Faks makinesi
-> PDA

1997
Nokia Communicator



Nokia
Akıllı telefon
-> LCD ekranı,
-> QWERTY klavye

Beşinci Nesil (1990-Günümüz): Mobil Bilgisayarlar

2007

Apple iPhone



Apple
Dokunmatik ekran
3G

2008

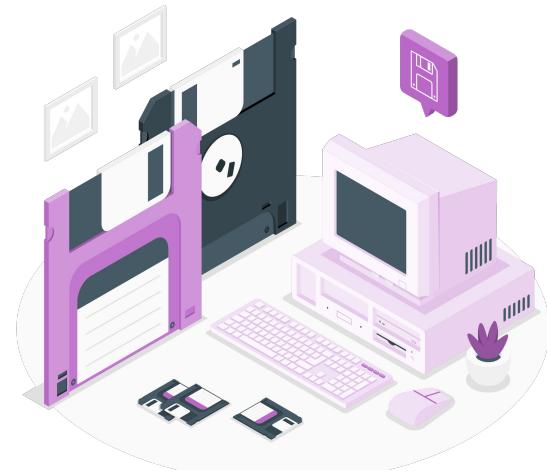
HTC Dream



HTC
Android OS

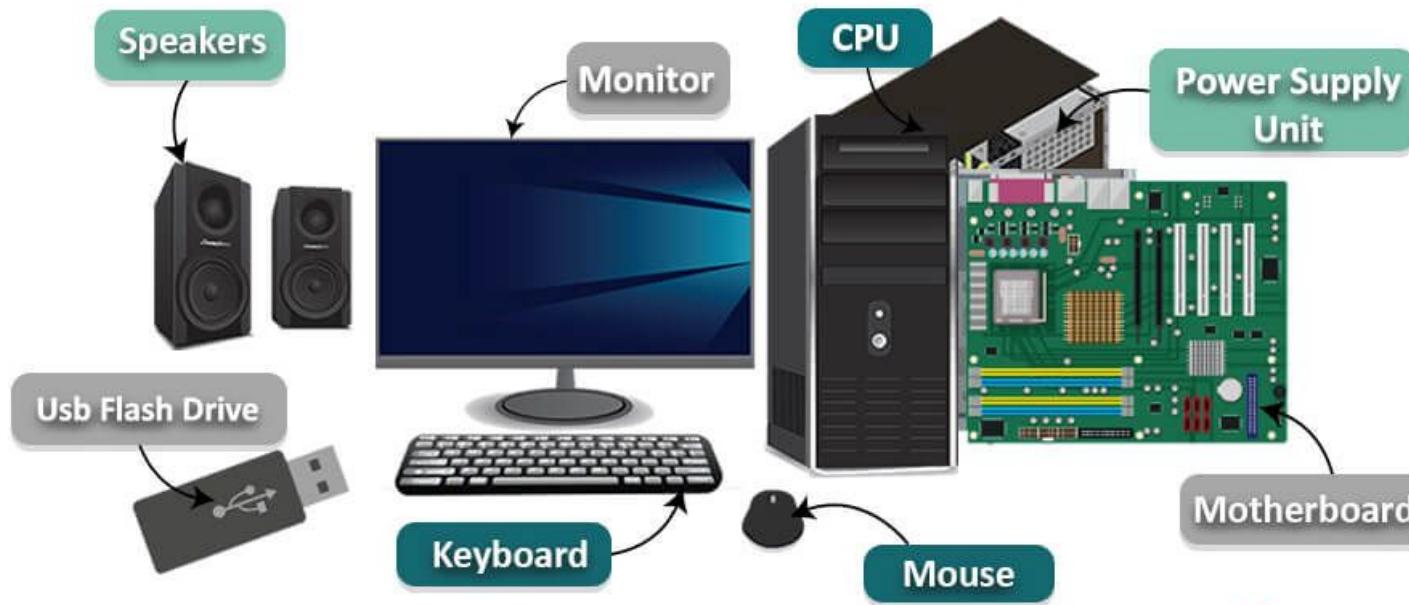
04

Bilgisayar Donanımı

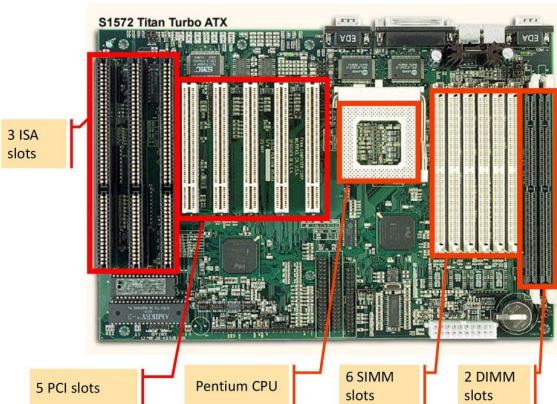


Giriş

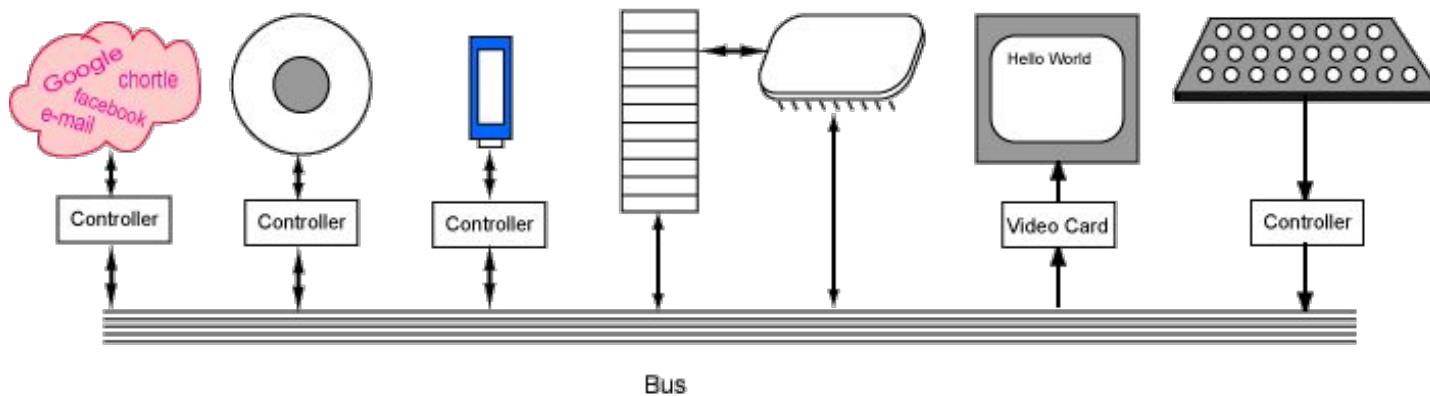
Components of Computers



Giriş

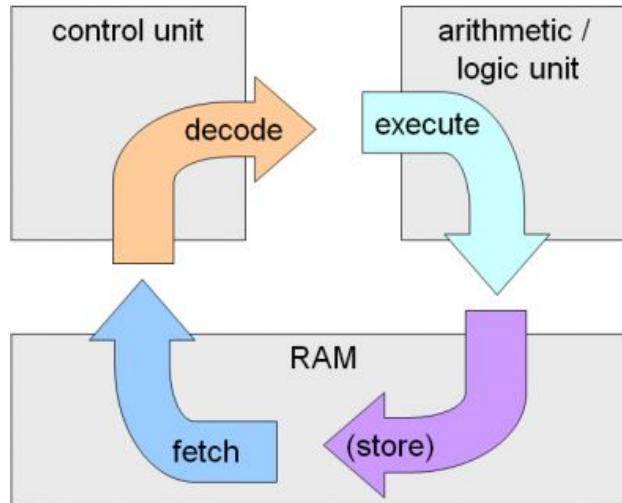
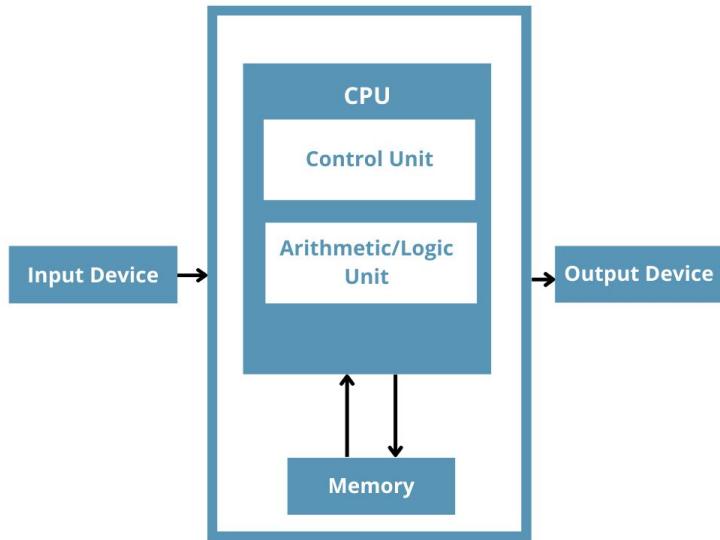
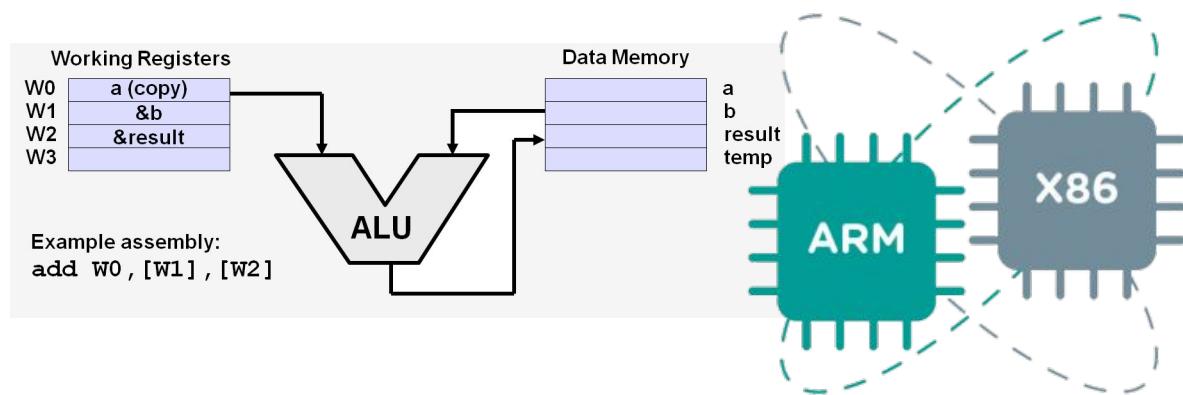


Network Hard Disk USB Drive Main Memory Processor Monitor Keyboard



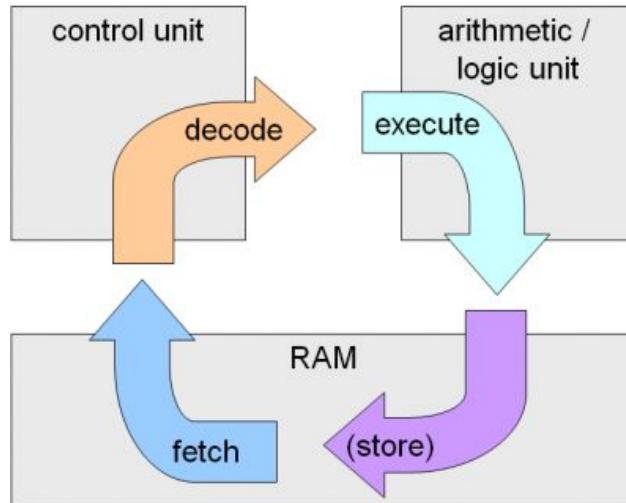
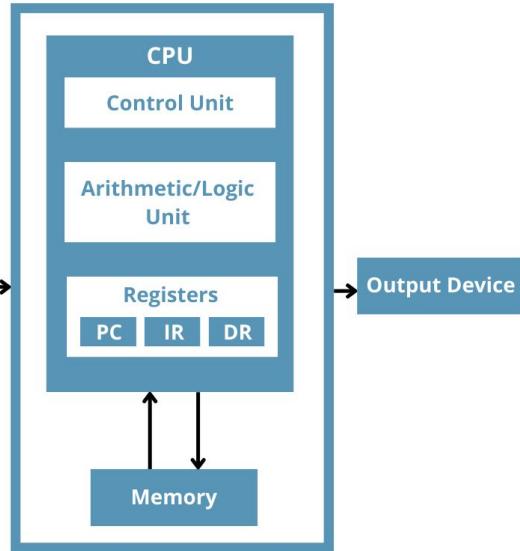
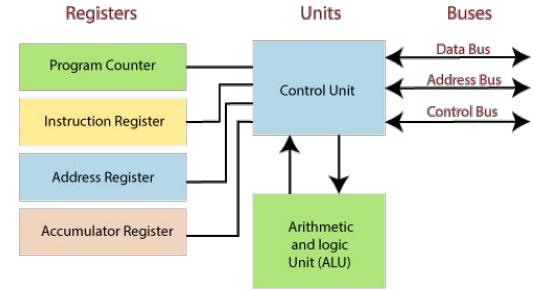
Main Components of a Computer System

İşlemci CPU

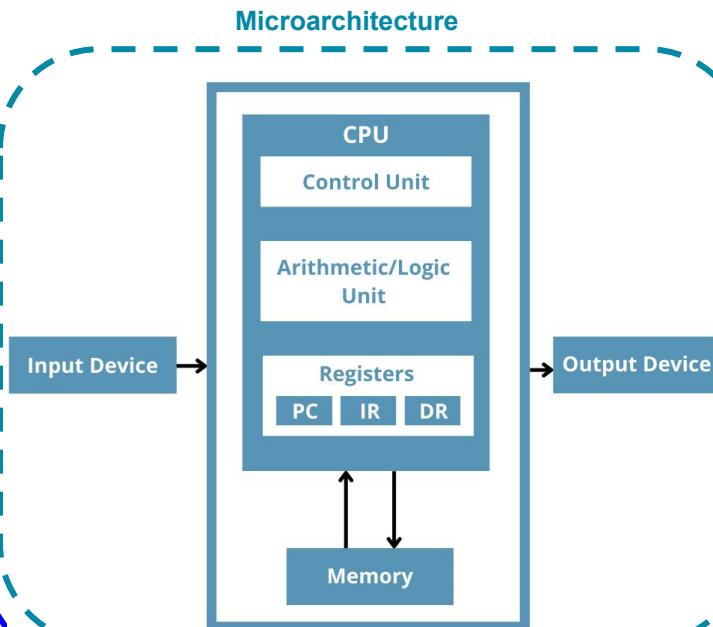


İşlemci CPU

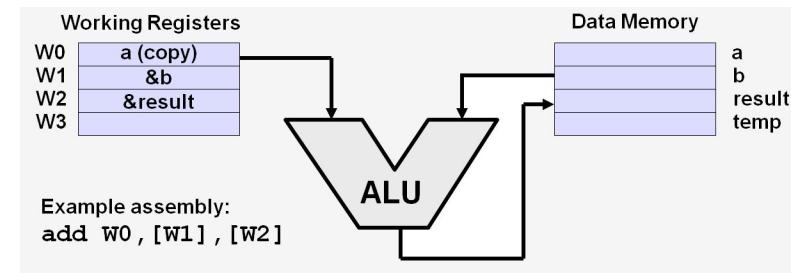
$t_{\text{fetch}} > t_{\text{execute}}$



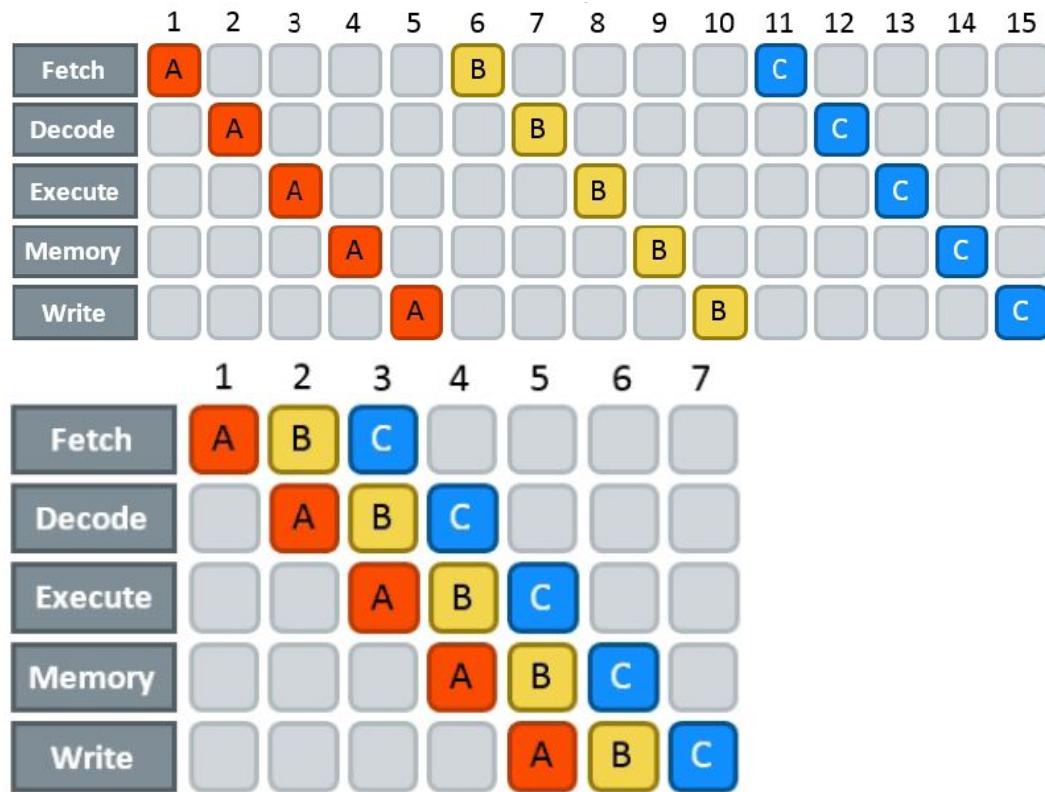
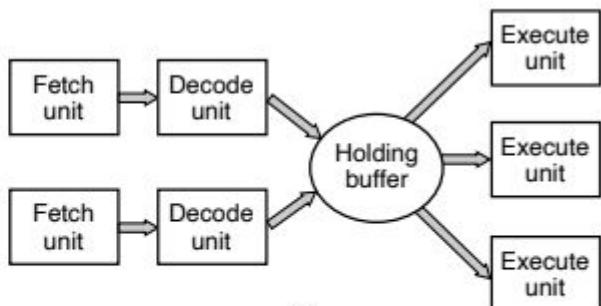
İşlemci CPU



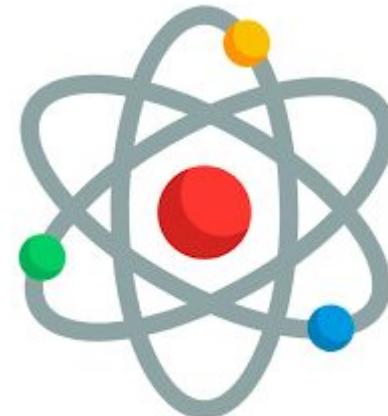
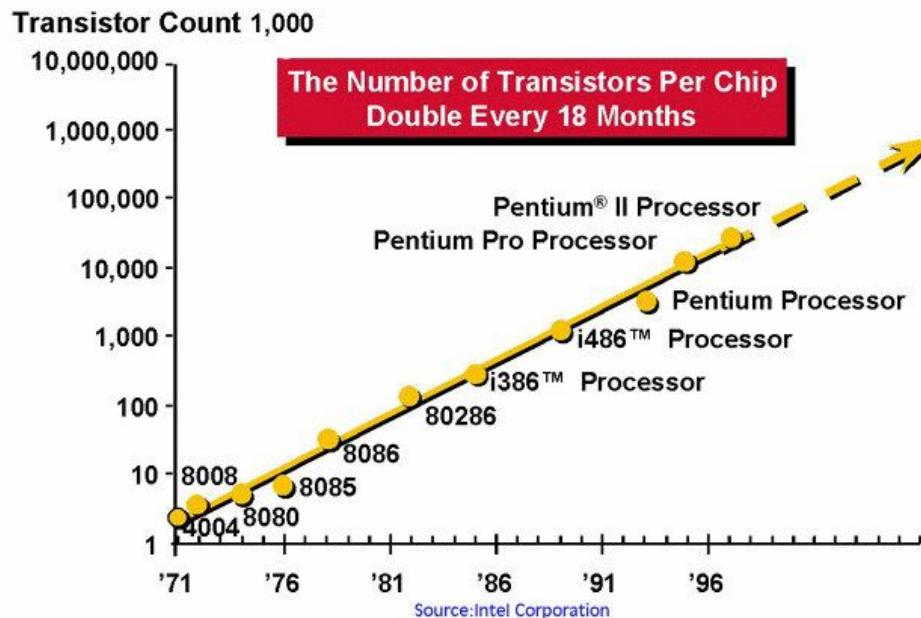
Architecture



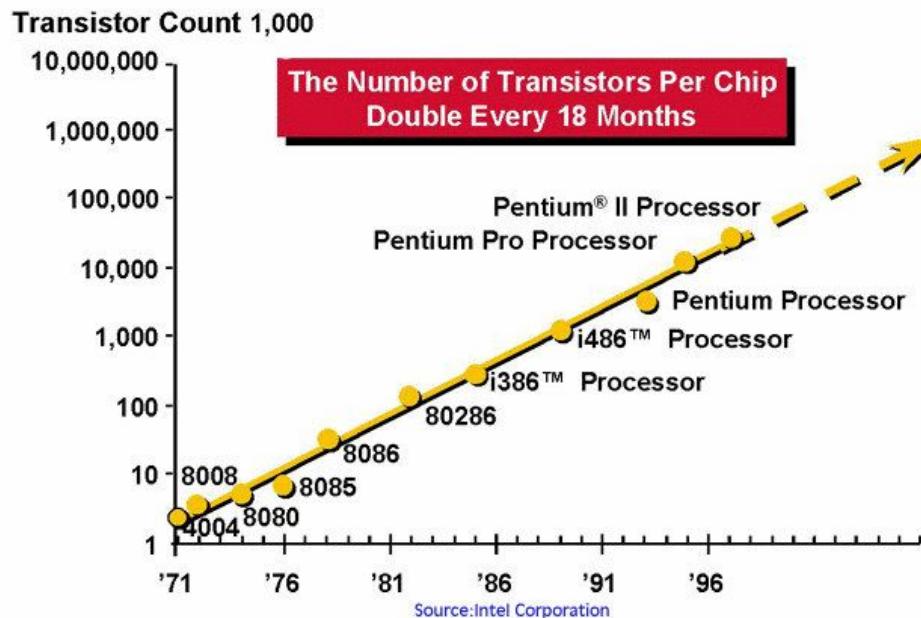
İşlemci CPU



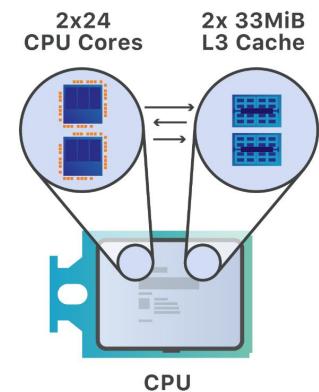
Çok İş Parçacıklı ve Çok Çekirdekli CPU



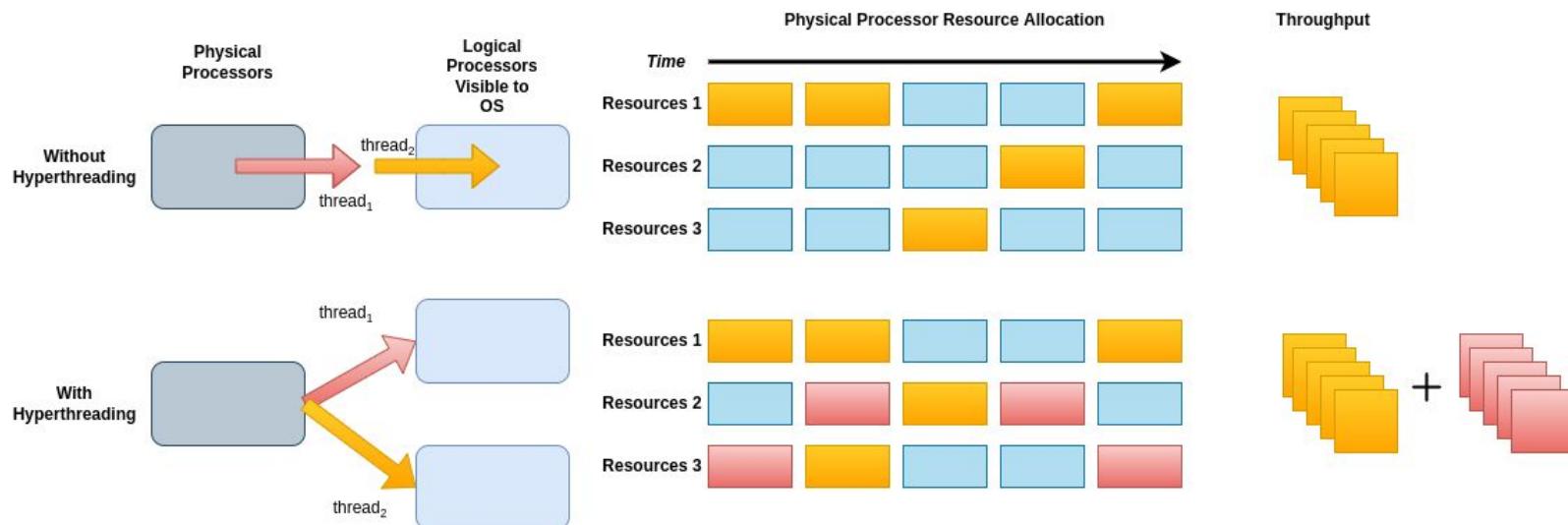
Çok İş Parçacıklı ve Çok Çekirdekli CPU



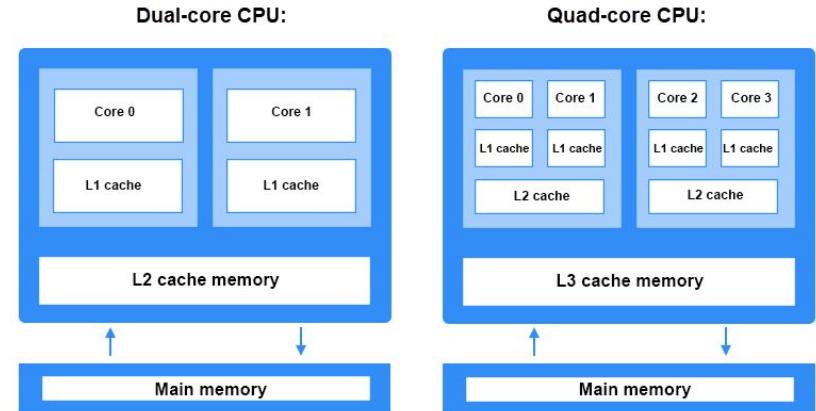
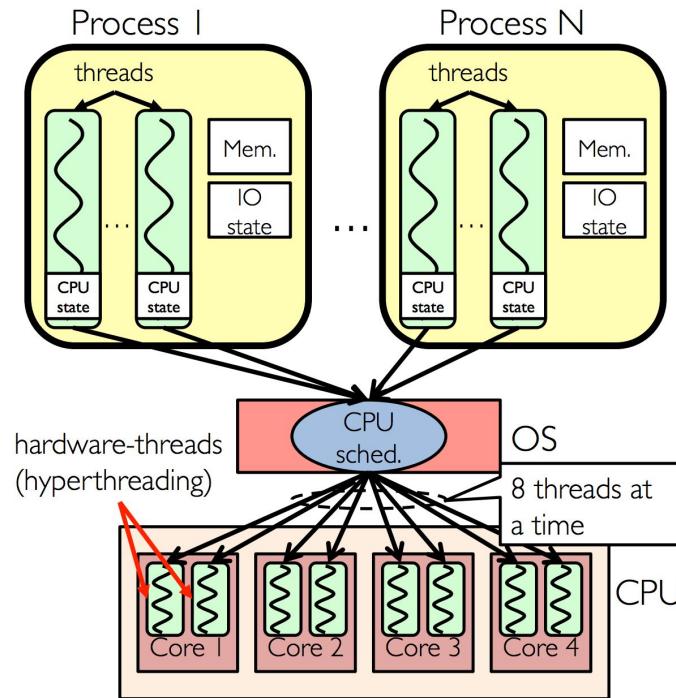
bigger



Çok İş Parçacıklı ve Çok Çekirdekli CPU

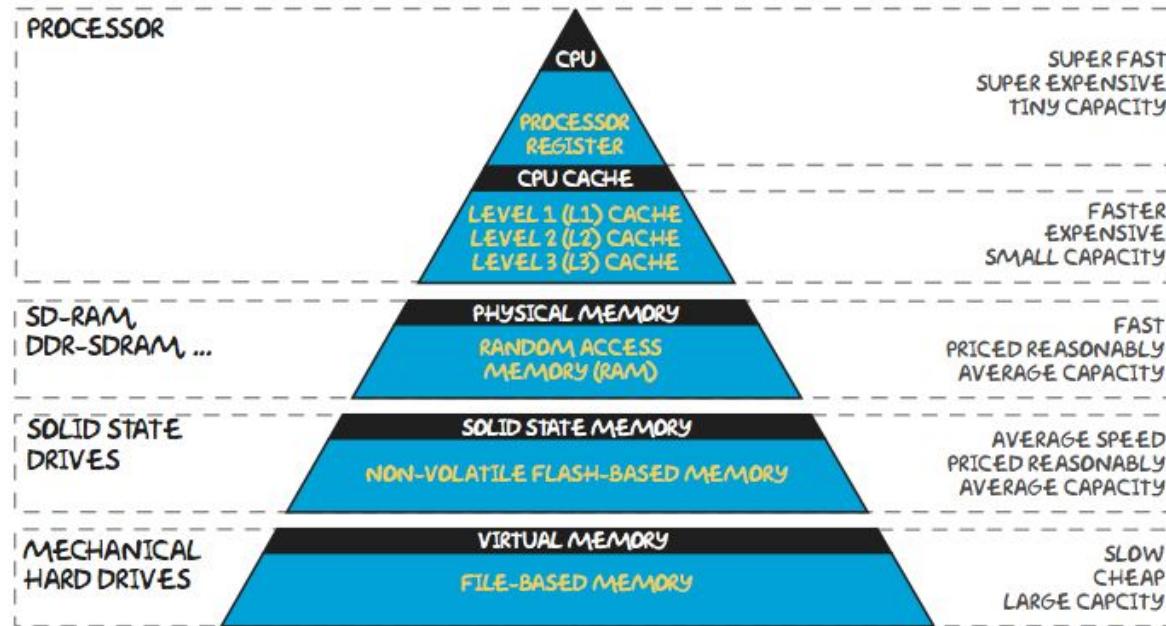


Çok İş Parçacıklı ve Çok Çekirdekli CPU



Bellek

THE MEMORY HIERARCHY



Bellek

32 bit CPU 32×32 bit

64 bit CPU 64×64 bit

THE MEMORY HIERARCHY



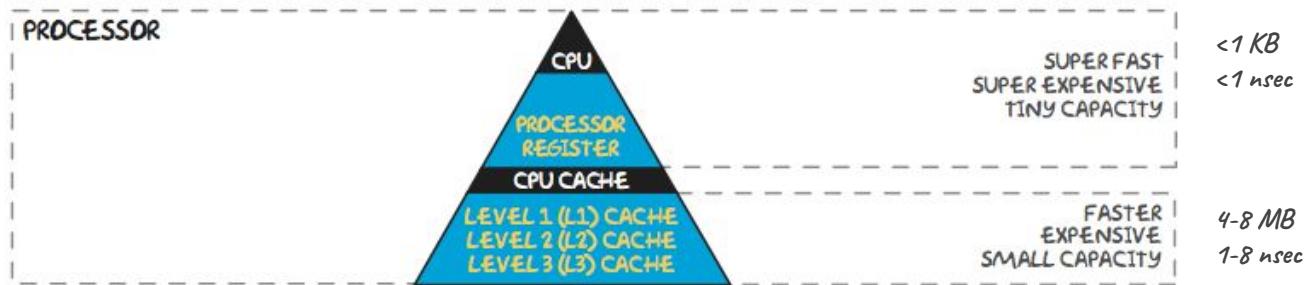
Bellek

32 bit CPU 32×32 bit
64 bit CPU 64×64 bit

0. satır 0 & 63 arasındaki adresler
1. satır 64 & 127 arasındaki adresler
....

Yoğun olarak kullanılan dosyalar,
dosya yolları (`/home/projects/minix3/src/kernel/clock.c`),
web adresi (URL -> IP)
vb.

THE MEMORY HIERARCHY



Önbelleğe yeni kayıt ekleme;

1. Ne zaman eklenecek
2. Hangi satırda yerleştirilecek
3. Hangi kayıt çakarılacak
4. Çıkarılan kaydın nereye yerleşeceği

L1 önbellek:

Yürüttülecek kodu gözülmüş talimatları tutar
Çok yoğun kullanılan veri sözcükleri tutar
32 KB boyutundadır

L1 önbelleğe erişim: Gecikmesiz

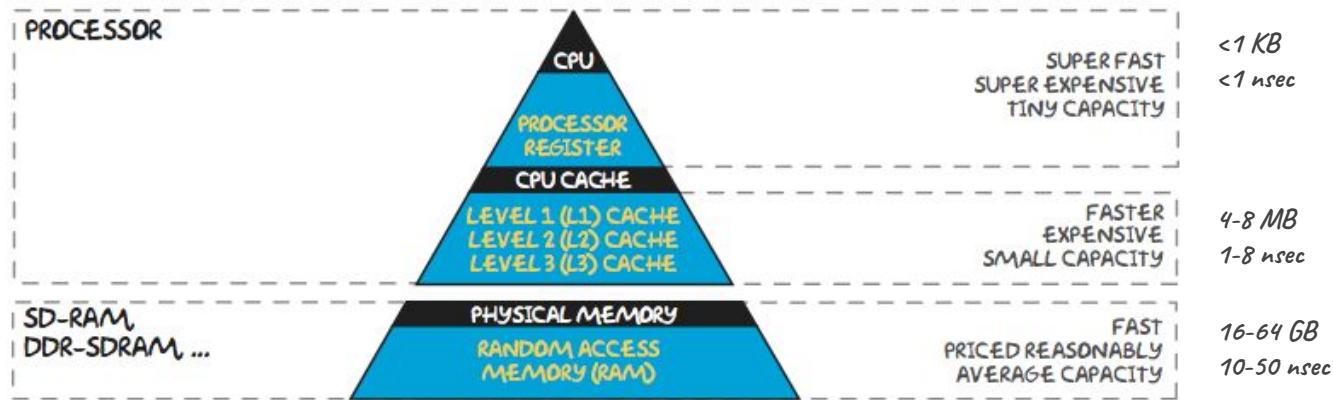
L2 önbelleğe erişim: Birkaç saat döngüsü gecikme

Bellek

THE MEMORY HIERARCHY

32 bit CPU 32×32 bit
64 bit CPU 64×64 bit

0. satır 0 & 63 arasındaki adresler
1. satır 64 & 127 arasındaki adresler
....



Önbellekten karşılanamayan tüm CPU istekleri ana belleğe gider.

ROM (Salt Okunur Bellek): Programlandıktan sonra değiştirilemez. Hızlı ve ucuzdur. bootstrap yükleyicisi ROM'da bulunur.

RAM (Random Access Memory):

EEPROM(Electronically Erasable Read-Only Memory): Uçucu değildir uzun süre de silinip yeniden yazılabilir.

Flash Bellek: Uçucu değildir ancak silinip yeniden yazılabilir. Hızlı bir alternatif olarak SSD'lerde kullanılır. Çok fazla kez silinirse yıpranır.

CMOS(Complementary Metal Oxide Semiconductor). Saati ve tarihi tutmak için ve yapılandırma parametreleri (hangi sürücüden önyükleme yapacağı) tutar.

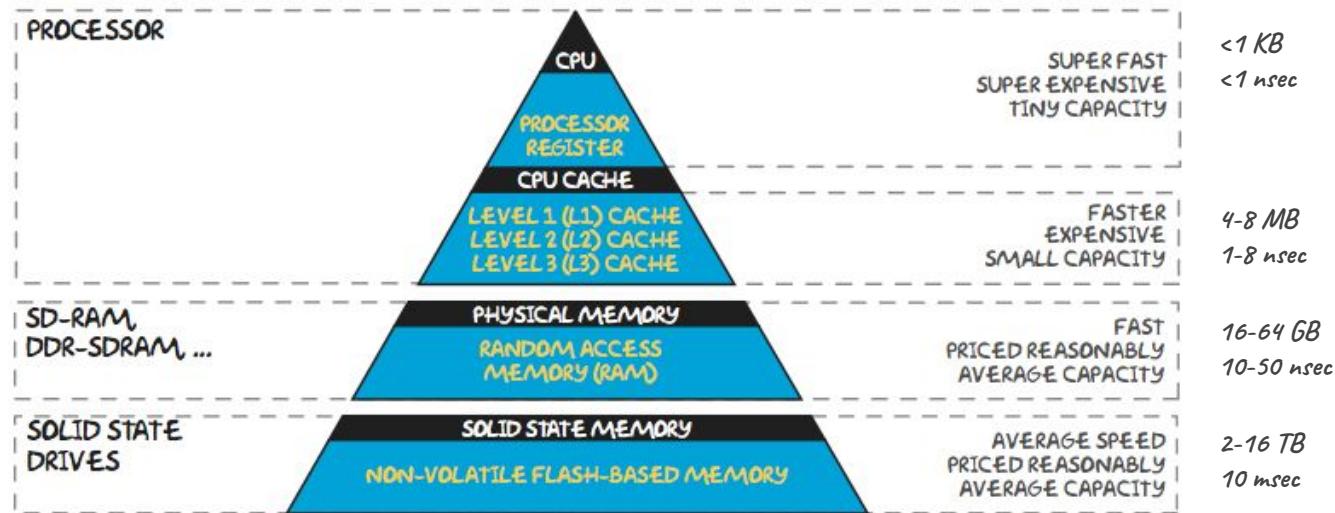
Sanal bellek sistemi: Ana belleği önbellek olarak kullanır. Sanal adresi, fiziksel adrese dönüştürmeyi CPU'daki MMU (Bellek Yönetim Birimi) yapar.

Bellek

32 bit CPU 32×32 bit
64 bit CPU 64×64 bit

0. satır 0 & 63 arasındaki adresler
1. satır 64 & 127 arasındaki adresler
....

THE MEMORY HIERARCHY



Fiziksel olarak disk olmamalarına (plakaları ya da hareketli kolları) rağmen SSD disk olarak kabul edilir.

Verileri Flash bellekte depolar. Uçucu değildirler.

Hızlı olmalarına rağmen bayt başına maliyet açısından dönen disklerden çok daha pahalıdır.

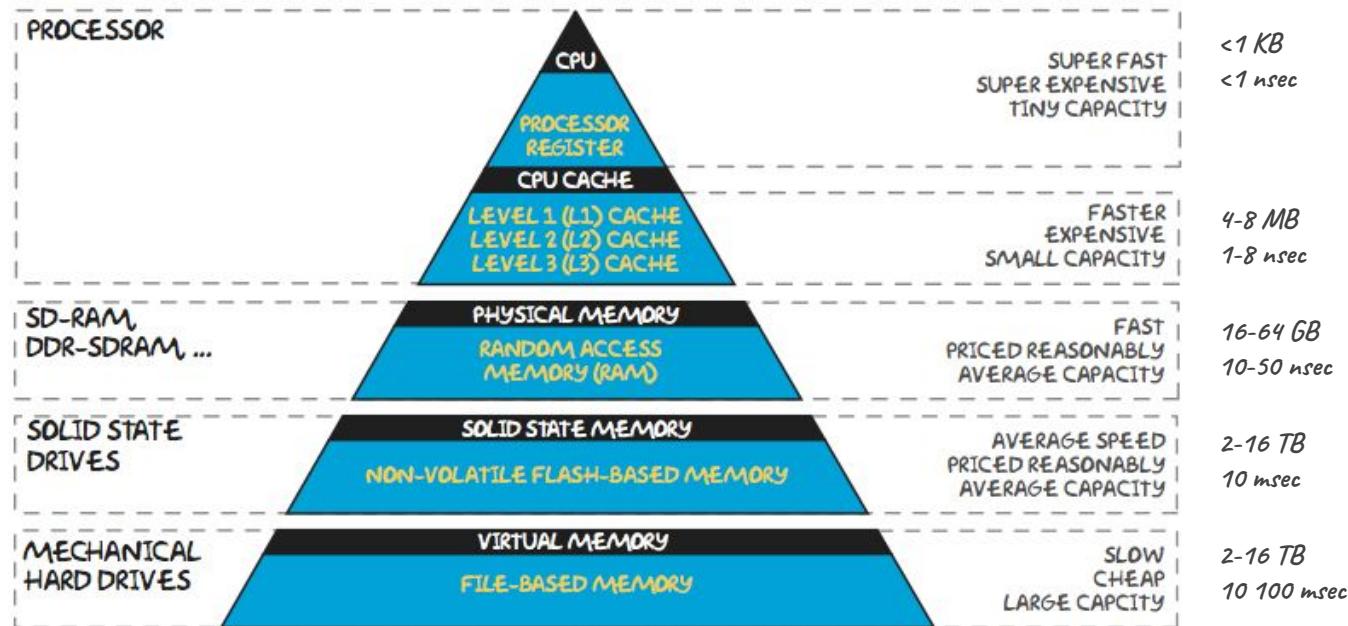
Mekanik bir kol olmadığından rastgele konumlardaki verilere erişebilirler.

Bellek

32 bit CPU 32×32 bit
64 bit CPU 64×64 bit

- 0. satır 0 & 63 arasındaki adresler
- 1. satır 64 & 127 arasındaki adresler
- ...

THE MEMORY HIERARCHY



Bellek

Disk (5400, 7200, ... RPM) hızında dönen plakalardan oluşur.

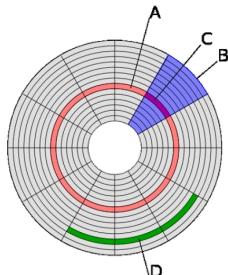
Mekanik bir kol plakaların üzerinde döner.

Bilgi disk üzerine daire şeklinde yazılır. Mekanik kol ucundaki kafa iz olarak bilinen dairesel bir bölgeyi okuyabilir.

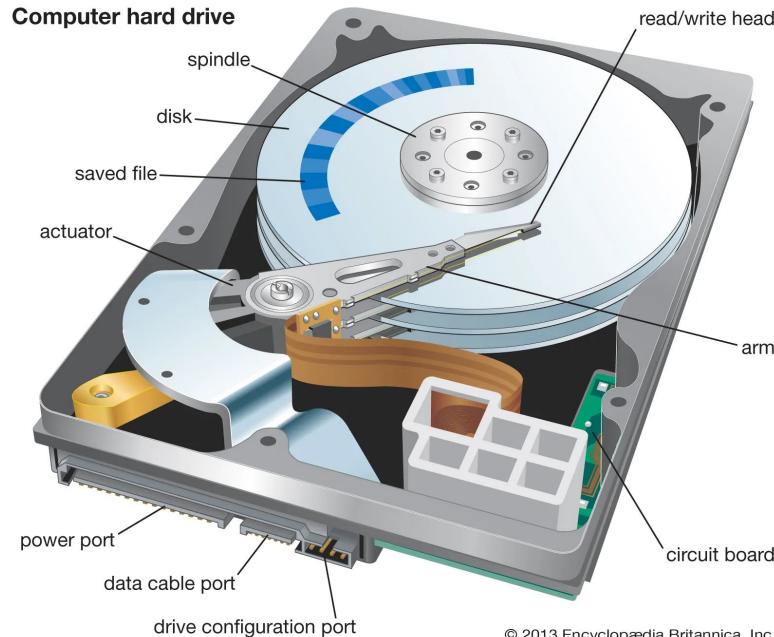
İzler, saniye başına belli sayıda sektör'e bölünmüştür.

Kolu bir silindirden diğerine taşımak yaklaşık 1 msn sürer.

Kol, sürücünün sektörün kafanın altında dönmesini bekler; gecikmeye (5-10 msn) neden olur.



Hard Drive Structure:
A = track
B = sector
C = sector of a track
D = cluster



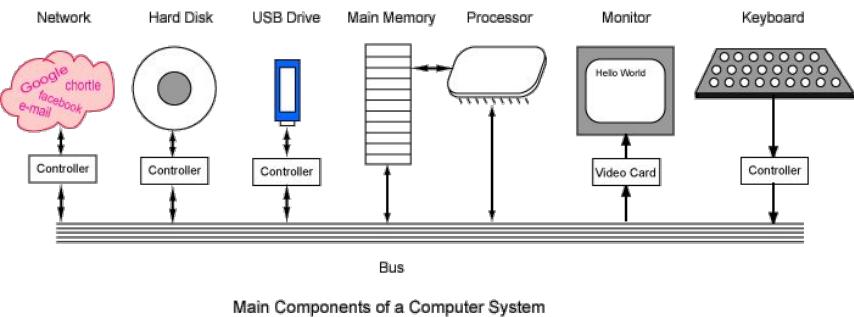
© 2013 Encyclopædia Britannica, Inc.

I/O Cihazları

SATA



SATA Denetleyici



Cihazın kontrolü karmaşık ve ayrıntılıdır, işletim sisteme daha basit arayüz sunmak
2. disktten 11,206. sektörü okuma komutunu alan denetleyici :

Sektör numarasını silindire, sektörre ve kafaya dönüştür

Disk kolunun hangi silindirde olduğunu belirler ve hedef sektörde yönlendirir

Hedef sektör kafanın altında dönene kadar beklemeler

Sürücüden alınan bitleri okumaya ve depolamaya başlar

Gelen bitleri kelime halinde bir araya getirir ve bellekte saklamalar.

Cihazlar standart olmaları için oldukça basit ara yüzlere sahiptir.

SATA: Serial ATA

ATA: AT Attachment

AT: IBM'in 1984 yılında tanıttığı "Personal Computer Advanced Technology"

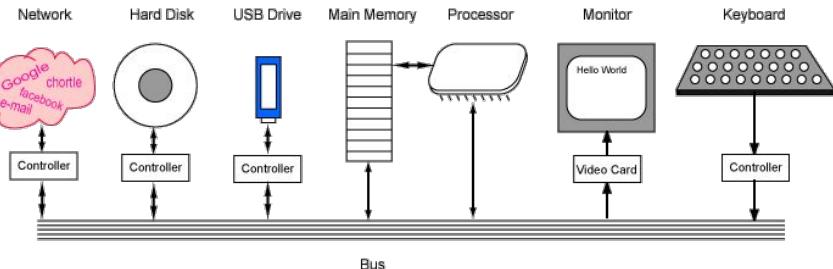
Gerçek aygit arayüzü denetleyicinin arkasında gizlendiğinden, işletim sisteminin gördüğü tek şey denetleyicinin arayüzüdür ve bu da aygitin arayüzünden oldukça farklı olabilir.

I/O Cihazları

SATA

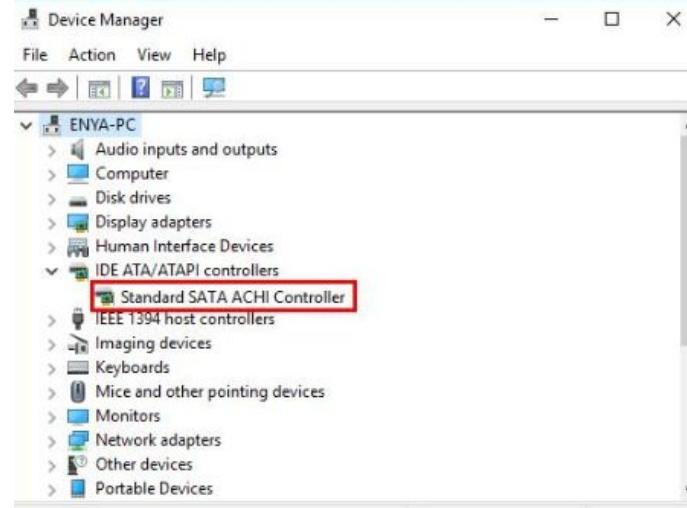


SATA Denetleyici

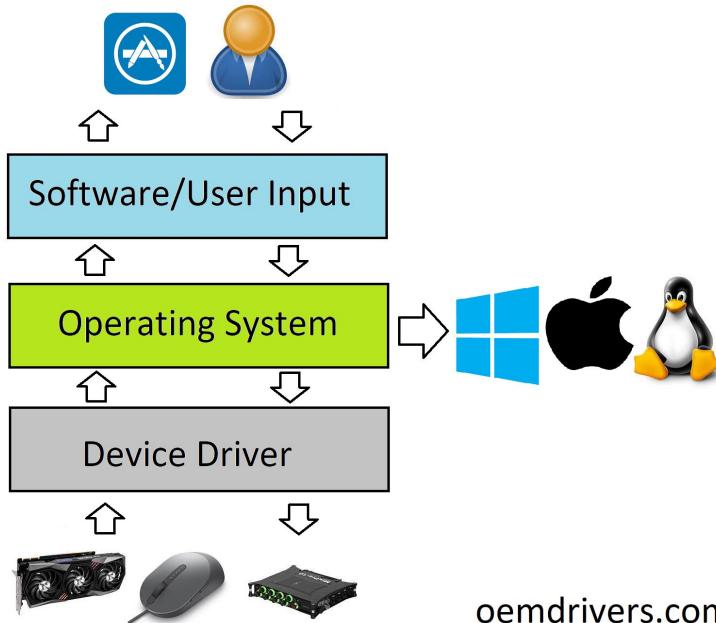


Main Components of a Computer System

SATA Windows Sürücü



I/O Cihazları

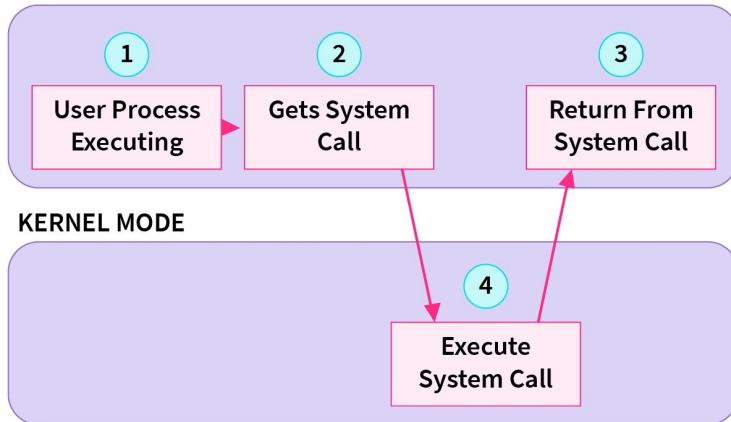


Sürücünün çekirdeğe yerleştirilmesinin üç yolu vardır.

1. Çekirdeğe yeni sürücüyü bağlamak ve sistemi yeniden başlatmaktır.
2. İşletim sistemi dosyasına sürücüye ihtiyaç duyduğunu belirten bir giriş yapmak ve sistemi yeniden başlatmaktır. Önyükleme sırasında işletim sistemi gidip ihtiyaç duyduğu sürücülerini bulur ve yükler.
3. İşletim sisteminin yeni sürücülerini galırken kabul edebilmesi ve yeniden başlatmaya gerek kalmadan anında yükleyebilmesidir.

I/O Cihazları

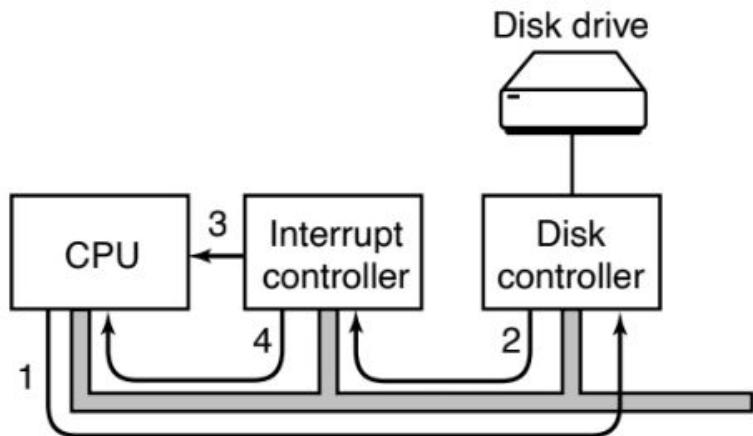
USER MODE



Giriş ve çıkış yöntemi (Meşgul Bekleme)

1. Kullanıcı programı bir sistem çağrıyı yayınlar
2. Çekirdek bunu uygun sürücü için bir prosedür çağrısına çevirir.
3. Sürücü daha sonra G/Ç'yi başlatır
4. Sıkı bir denetim döngüsü içinde işlemin tamamlandığını kontrol için cihazı sürekli olarak yoklanır (genellikle cihazın hala meşgul olduğunu gösteren bir bit vardır).
5. G/Ç tamamlandığında, sürücü verileri ilgili saklayıcılara yazar
6. İşletim sistemi daha sonra kontrolü çağrıya geri verir.

I/O Cihazları



Giriş ve çıkış yöntemi (Kesme)

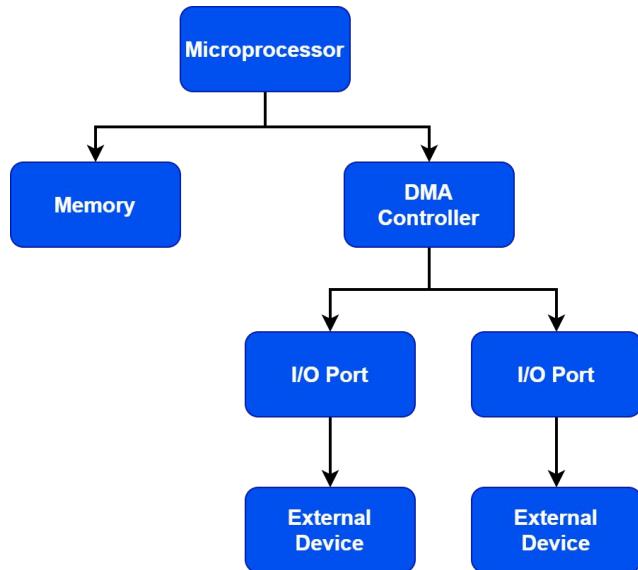
Adım 1-) Sürücü, aygit saklayıcılarını kullanarak denetleyiciye ne yapması gerektiğini söyler. Denetleyici aygıtını başlatır.

Adım 2-) Aygit denetleyicisi, işini bitirdiğinde, kesme denetleyicisine sinyal gönderir.

Adım 3-) Kesme denetleyicisi kesmeyi kabul ederse (daha yüksek öncelikli bir kesmeyi işlemekle meşgulse hazır olmayıabilir), CPU üzerindeki bir pinin uyararak bunu bildirir.

Adım 4-) Kesme denetleyicisi cihazın numarasını CPU bildirir

I/O Cihazları



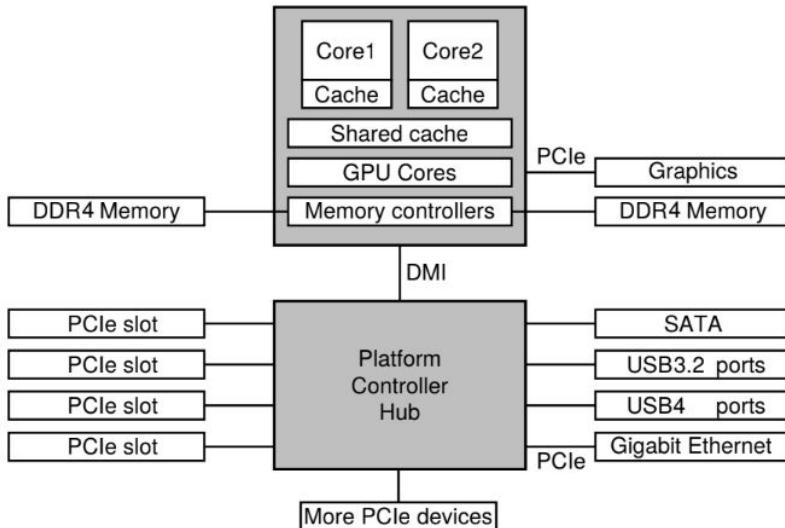
Giriş ve çıkış yöntemi (DMA Direct Memory Access)

Denetleyiciler arasındaki bit akışını kontrol eden DMA (Doğrudan Bellek Erişimi) yongası

CPU, DMA yongasını kurar (kaç bayt aktarılacağını, ilgili aygit ve bellek adreslerini bildirir)

DMA çipi işini bitirdiğinde, kesmeye tetikler.

Yollar



x86 Mimarisi

X86 Mimarisinde farklı aktarım hızına ve işleve sahip veri yolları vardır.

(PCI, USB, SATA ve DMI vb.) İşletim sistemi bu veri yollarının tümünü yönetmektedir.

Ana veri yolu PCIe (Peripheral Component Interconnect Express) veri yoludur.

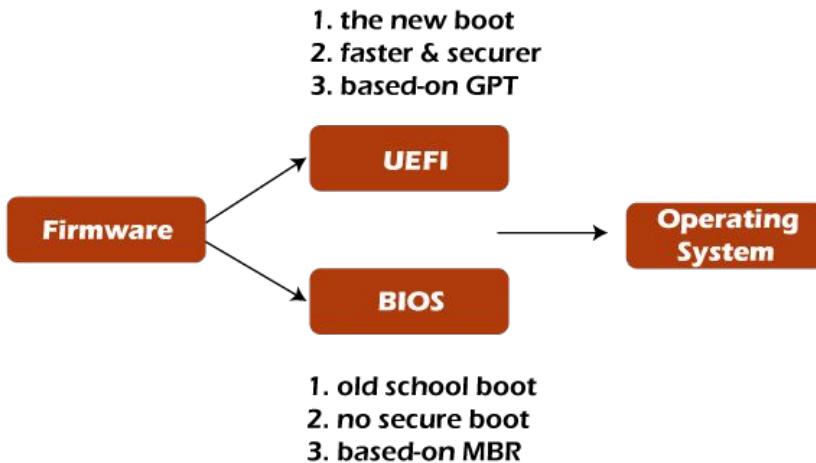
Intel tarafından eski PCI veri yolu'nun halefi olarak icat edilmiştir

Saniyede onlarca gigabit aktarma kapasitesine sahiptir

PCIe (2004) kadar, çoğu veri yolu paralel ve paylaşımlıydı.

PCIe özel, noktadan noktaya bağlantılar kullanır.

Önyükleme



Anakartdaki flaş diskte, BIOS (Temel Giriş Çıkış Sistemi) yazılımı barındırmaktadır.
BIOS kullanan önyükleme yapmak:

Yavaş önyükleme

Mimariye bağımlı

Küçük SSD'ler ve disklerle (2 TB'a kadar) sınırlı

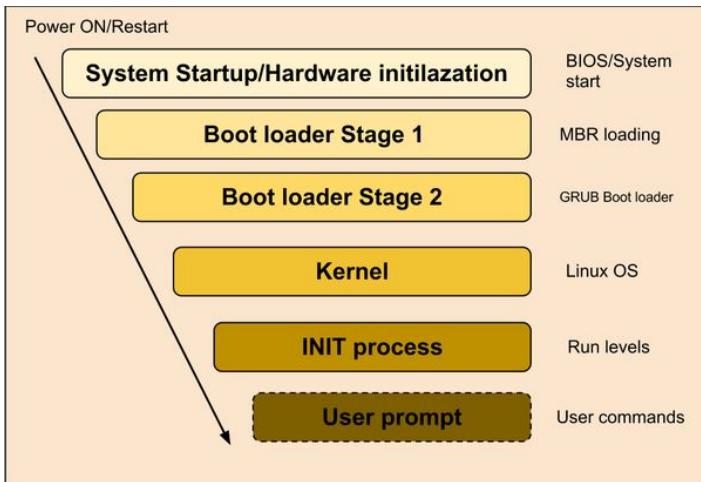
Intel, UEFI'yi (Unified Extensible Firmware Interface - Birleşik Genişletilebilir Ürün Yazılımı Arayüzü)

Hızlı önyüklemeye

Farklı mimarilere çalışma

8 ZiB kadar depolama

Önyükleme



CPU, başlatma kodunu anakarttaki flashta bulunan sabit bir adresten (sıfırlama vektörü olarak bilinir) alır.

Başlatma kodu, RAM, Platform Denetleyici Hub, PCI/ PCIe veri yollarına bağlı aygıtları tespit eder ve başlatır.

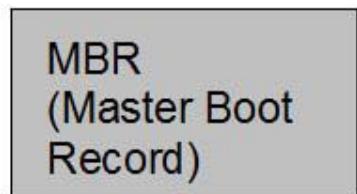
BIOS, CMOS belleğindeki listeden önyükleme aygıtını belirler.

Kullanıcı, önyüklemeden sonra BIOS programına girerek bu ön yükleme aygit listesini değiştirebilir.

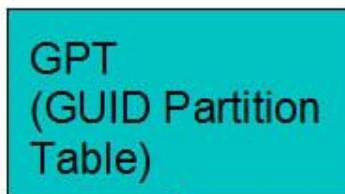
- 1-) Önyükleme aygıtındaki ilk sektör MBR, ikincil önyüklemeyi aktif eder
- 2-) ikincil önyükleme yükleyicisi işletim sistemini başlatır.
- 3-) İşletim sistemi BIOS'tan aygit sürücülerini sorgular ve gerekirse yükler.
- 4-) Arka plan işlemlerini çalıştırarak bir oturum açma programı veya GUI başlatır.

Önyükleme

BIOS



UEFI



1-) Önyükleme aygıtının ikinci sektöründeki GPT (GUID Bölümleme Tablosunun) yerini arar.

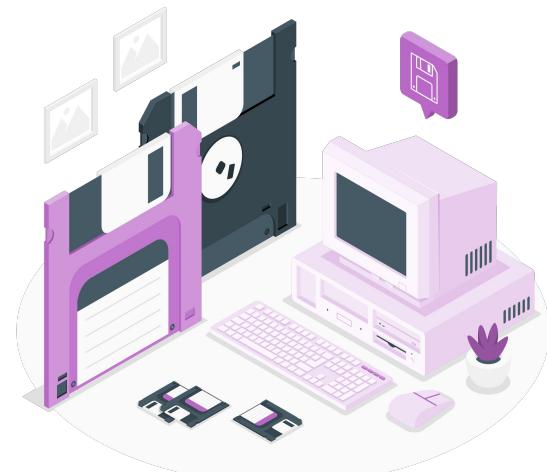
2-) UEFI tarafından desteklenen dosya sistemlerinden biri EFI sistem bölümü (ESP) olarak bilinen özel bir bölüme yerleştirilir.

3-) Önyükleme işlemi artık programları, yapılandırma dosyalarını ve önyükleme sırasında yararlı olabilecek diğer her şeyi içeren uygun bir dosya sistemi kullanabilir.

UEFI, bölümleri, dosya sistemlerini, çalıştırılabilir dosyaları vb. anlayan küçük bir işletim sistemine bezenilebilir

05

Sistem Çağrıları



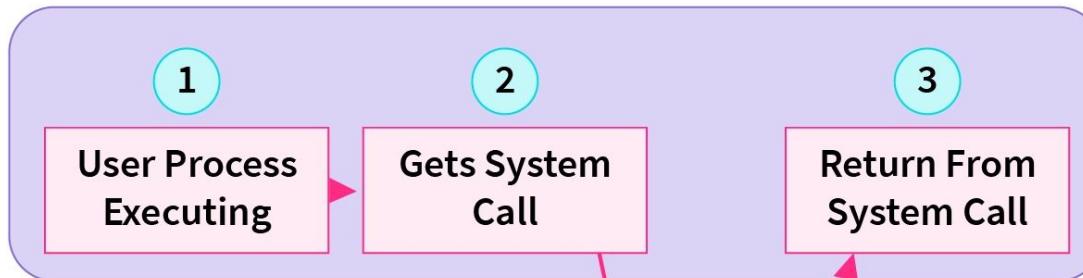
Giriş



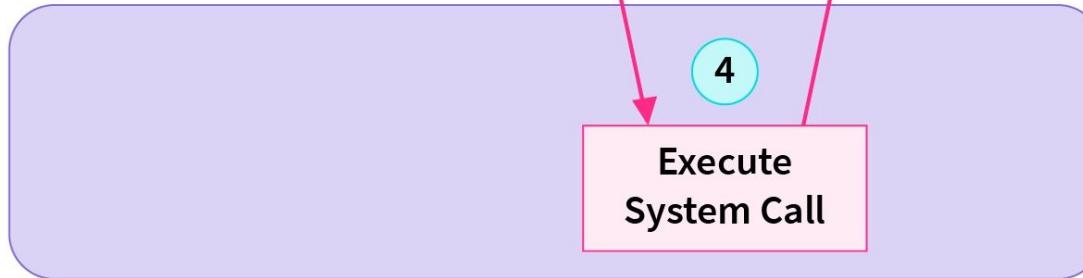
Giriş

WORKING OF A SYSTEM CALL

USER MODE



KERNEL MODE



Giriş

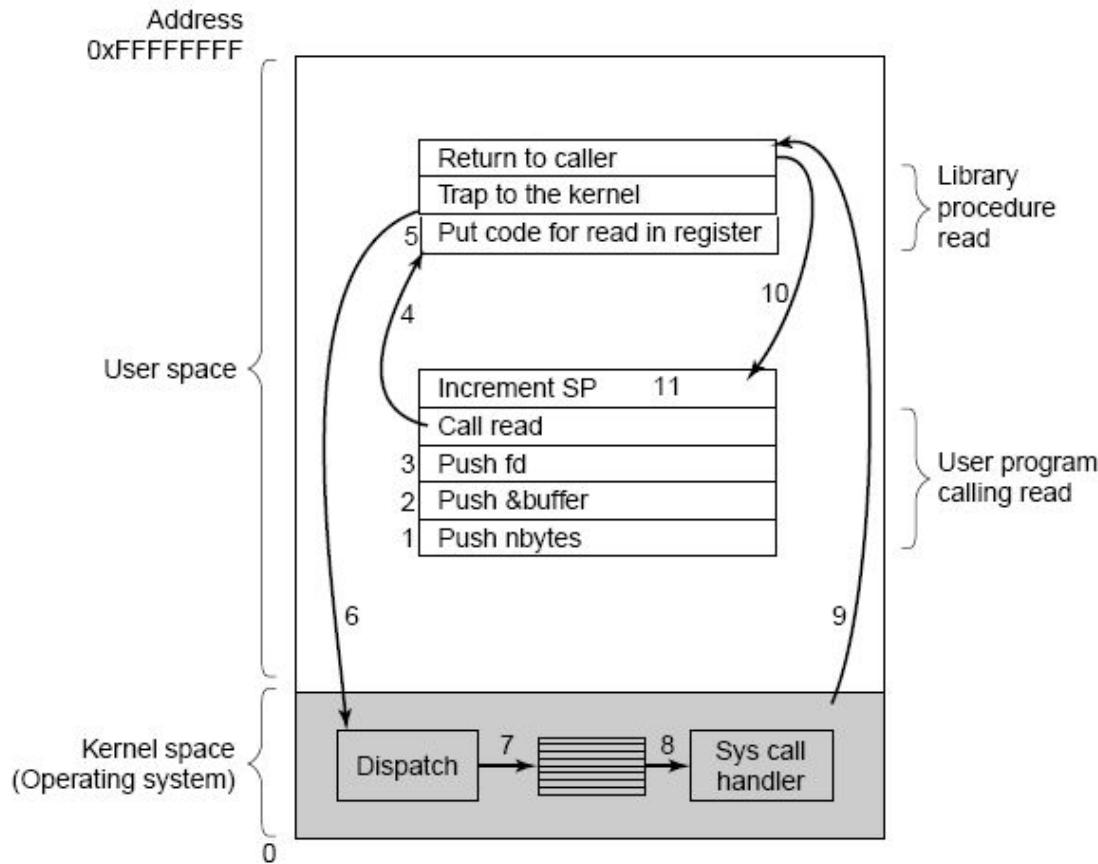
```
count = read(fd, buffer,  
nbytes);
```

count = bayt sayısını

Bu değer nbytes ile aynıdır, ancak okuma sırasında dosya sonu ile karşılaşılırsa daha küçük olabilir.

Sistem çağrısı gerçekleştirilemezse, count < 1

Hata numarası global değişkene (errno) konur.



Giriş

```
count = read(fd, buffer,  
nbytes);
```

1 ve 3-> Parametreler aktarımı

2 -> Tampon adresi aktarımı

4-> Kütüphane prosedürüne yapılan çağrı

5-> Sistem çağrı numarasını saklayıcıya (RAX) konur

6-> Sabit bir adreste yürütmemeyi başlatmak için tuzak komutu çalıştırılır

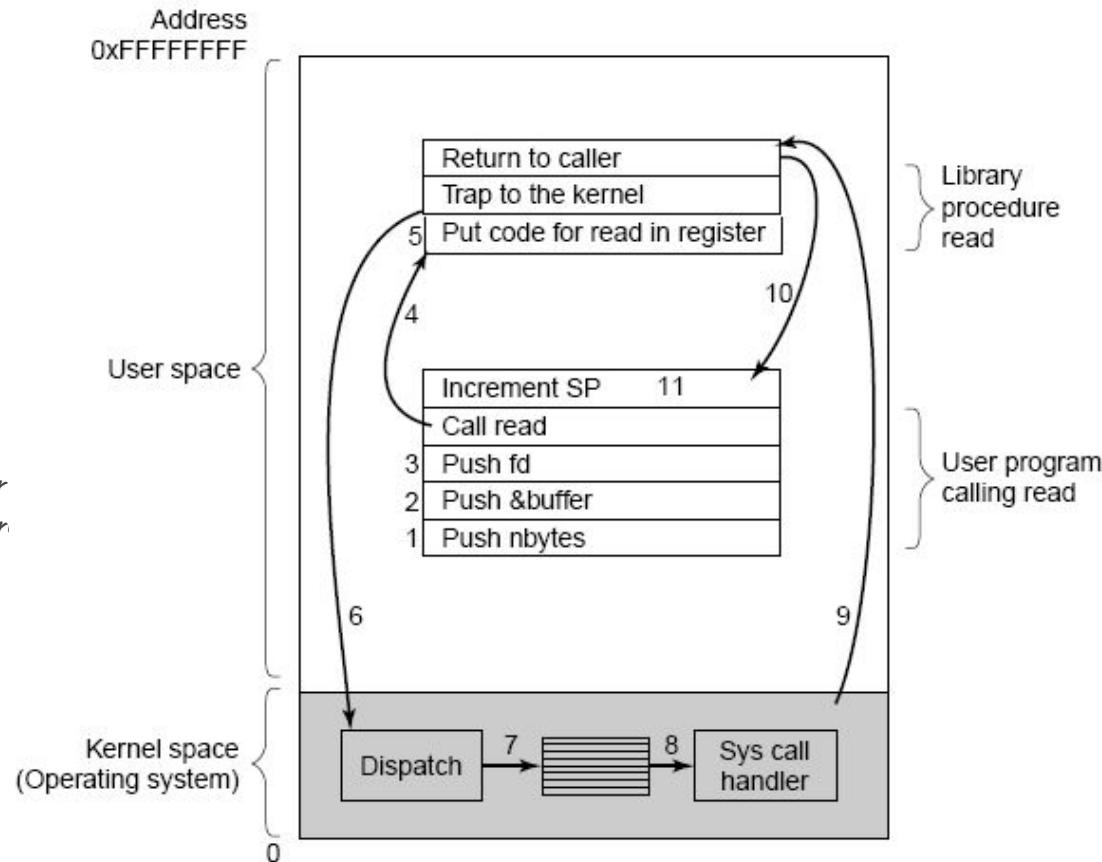
7-> İstek, sistem çağrı numarasına göre sistem çağrı işleyicisine gönderilir

8-> Sistem çağrı işleyicisi çalışır

9-> Kontrol kütüphane prosedürüne geri döndürülür

10-> Kendini çağrıran prosedüre döner

11-> Bir sonraki komutu çalıştırır



Süreç Yönetimi için Sistem Çağrıları

Process management

Call	Description
pid = fork()	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

Fork -> Yeni bir süreç yaratma

Orijinal sürecin tam bir kopyasını oluşturur.

Orijinal süreç ve kopya (ebeveyn ve çocuk) bağımsızdır

Çocuğun belleği ebeveyn ile copy-on-write paylaşılabilir.

Çocukta sıfır, ebeveynde çocuğun PID'sine (Process IDentifier) eşit olan bir değer döndürür.

Süreç Yönetimi için Sistem Çağrıları

fork, waitpid ve execve(exec1, execv, execle, execve) kullanımı

execve üç parametreye sahiptir:

1. Çıktırılacak dosyanın adı,
2. Argüman dizisini tutan işaretçi
3. Ortam dizisini tutan işaretçi

main(argc, argv, envp)

argc: program adı da dahil olmak üzere komut satırındaki öğelerin sayısı = 3

argv: dizi işaretisi; argv[0] =>"cp" argv[1] =>"file1" argv[2] =>"file2"

envp: terminal türü, ev dizini adı bilgilerini içeren ortamın bir göstericisidir.

Çocuk sürece hiçbir ortam aktarılmamıştır, bu nedenle execve'nin üçüncü parametresi sıfırdır.

```
#define TRUE 1
while (TRUE) {
    type_prompt();
    read_command(command, parameters);
    if (fork() != 0) {
        /* Parent code. */
        waitpid(-1, &status, 0);
    } else {
        /* Child code. */
        execve(command, parameters, 0);
    }
}
/* repeat forever */
/* display prompt on the screen */
/* read input from terminal */
/* fork off child process */
/* wait for child to exit */
/* execute command */
```

Dosya Yönetimi için Sistem Çağrıları

Dosyayı açan çağrı: O_RDONLY, O_WRONLY, O_RDWR

Dosya oluşturan çağrı: O_CREAT

Bir dosyanın herhangi bir bölümüne rastgele erişebilmesi gereklidir.

Konum işaretçisinin değerini değiştiren çağrı: lseek

Sistem çağrıları: Dosya tipini, boyutunu, son değişiklik zamanını tutan çağrı: stat

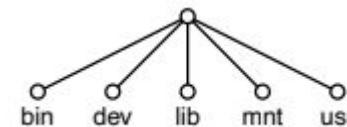


Dizin Yönetimi için Sistem Çağrıları

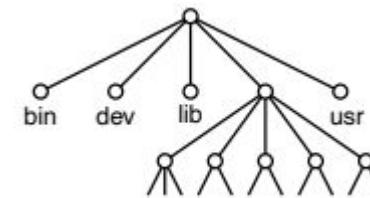
```
link("/usr/jim/memo", "/usr/ast/note");
```

/usr/ast		/usr/jim	
16	mail	31	bin
81	games	70	memo
40	test	59	f.c.
		38	prog1

```
mount("/dev/sdb0", "/mnt", 0);
```



/usr/ast		/usr/jim	
16	mail	31	bin
81	games	70	memo
40	test	59	f.c.
70	note	38	prog1



Windows API

- Windows, işletim sistemi hizmetlerini almak için kullanılan bir dizi prosedüre sahiptir (*Application Programming Interface: WinAPI, Win32 API, Win64 API*)
- Binlerce Win32 API (tüm sürümler ile uyumludur), sistem çağrılarının önemli bir kısmı kullanıcı alanında gerçekleştirilir. Dolayısıyla neyin sistem çağrıı neyin sadece kullanıcı uzayı kütüphane çağrıı olduğunu görmek imkansızdır.
- Win32 API, GUI özelliklerini yönetmek için çok sayıda çağrıya sahiptir.
- Windows'ta UNIX'teki gibi bir süreç hiyerarşisi yoktur, bir işlem oluşturulduğundan sonra, oluşturan ve oluşturulan eşittir.

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
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, so no umount
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

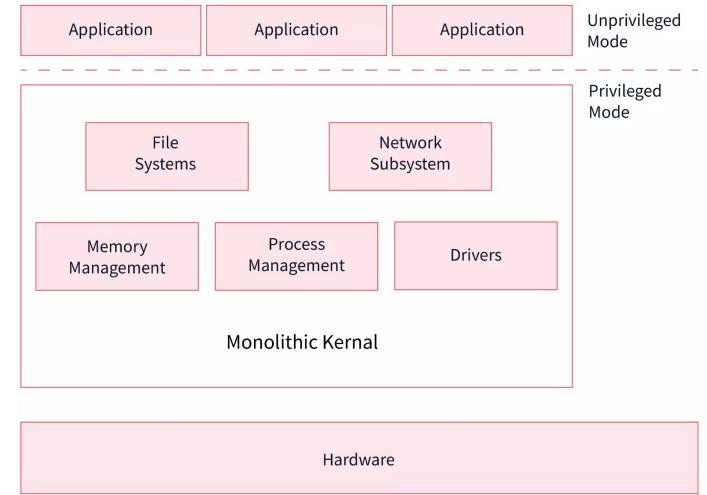
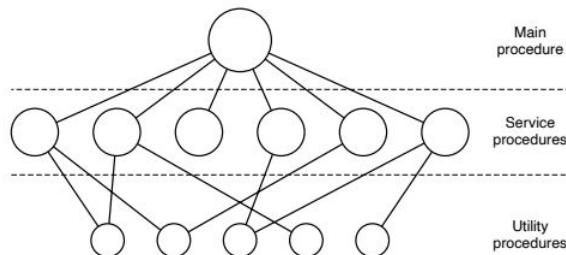
05

İşletim Sistemi Mimarisi



Monolitik Sistemler

1. Tüm işletim sistemini çekirdek modunda tek bir program olarak çalıştırmaktadır.
2. İşletim sisteminine ait prosedürler oluşturmak için önce tüm bireysel prosedürler derlenir ve ardından sistem bağlayıcısı kullanılarak hepsi tek bir çalıştırılabilir dosyada bir araya getirilir.
3. İşletim sistemi, birbirine bağlanmış bir prosedürler koleksiyonu olarak yazılır.
4. Sistemdeki her yordam, diğerini kolayca çağırabilmektedir.
5. Kısıtlama olmaksızın çağırabilen binlerce yordama hantal ve anlaşılması zor bir sisteme yol açar Prosedürlerden herhangi birinde meydana gelecek bir gökme tüm işletim sistemini çökecektir.
6. Her prosedür diğer tüm prosedürler tarafından görülebilir
7. Her sistem çağrısi için bir prosedür mevcuttur
8. Yardımcı prosedürler, hizmet prosedürleri tarafından ihtiyaç duyulan görevleri yürütür



Katmanlı Sistemler

Her biri bir altakinin üzerine inşa edilen altı katmanlardan oluşmaktadır.

Katman 0 -> İşlemci tahsisi, kesme yönetimi, eş zamanlı programa sürecinin yürütülməsi

Katman 1 -> Bellek yönetimi

Katman 2 -> Her bir işlem ile operatör konsolu (yani kullanıcı) arasındaki iletişimini

Katman 3 -> G/C cihazlarının yönetilmesi

Katman 4 -> Kullanıcı programlarının yönetimi

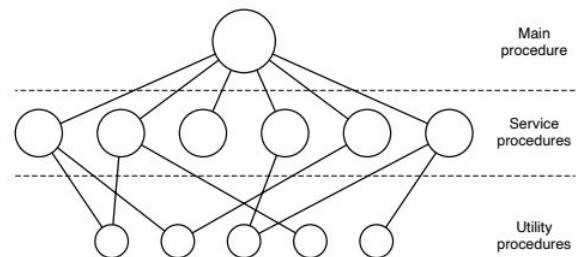
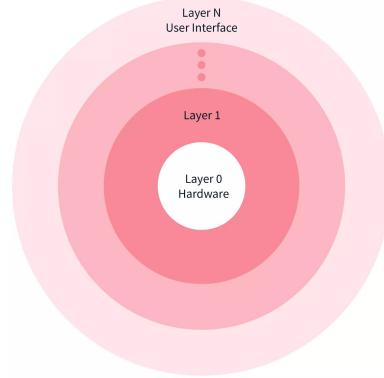
Katman 5 -> Kullanıcıların yer aldığı katman

Avantaj: Genişleme ve koruma

Bir profesör öğrencilerin yazdıkları programlara not vermek için kendi programını halkasında çalıştırır

Öğrenciler ise öğrenci programları $n + 1$ halkasında çalışır

Böylece notlarını değiştiremezler.



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

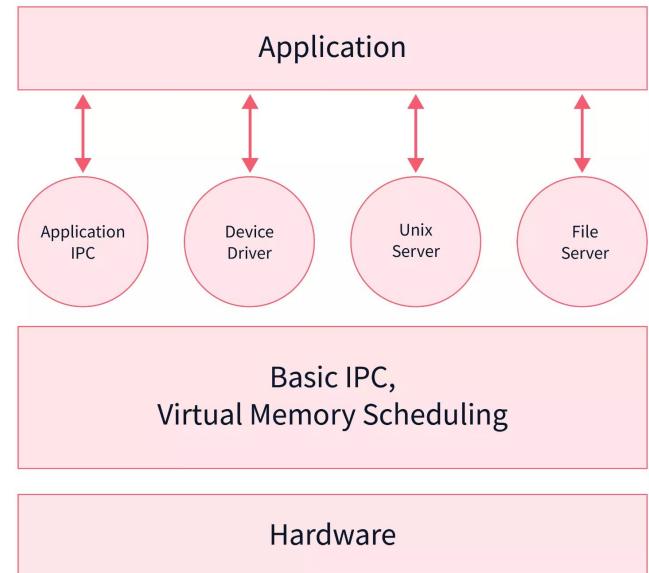
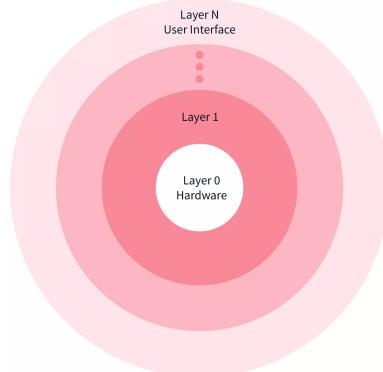
Mikroçekirdekli Sistemler

Katmanlı yaklaşımında, tüm katmanlar çekirdeğe yerleştirilir

Cekirdekteki hatalar sistemi anında çökertebilir.

1000 satır kod başına düşen hata sayısını = 2 ~ 10 hata

Monolitik işletim sistemi 5 milyon satır kod = 10.000 ~ 50.000 çekirdek hafası



Mikroçekirdekli Sistemler

İşletim sistemini küçük, iyi tanımlanmış modüllere ayırmak

Modüllerden yalnızca biri çekirdek modunda diğerleri kullanıcı modunda çalışır

Her bir aygıt sürücüsü ve dosya sistemi ayrı bir kullanıcı süreci olarak çalışır

Modüllerden birindeki hata o bileşeni gökertebilir, ancak tüm sistemi gökertemez

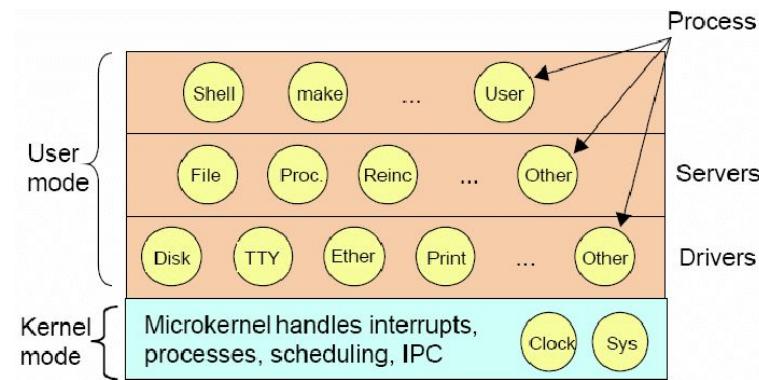
macOS haricinde, masaüstü işletim sistemleri mikro çekirdek kullanmaz

Kritik görevli, yüksek güvenilirlik gerekliliği zamanlı uygulamarda (endüstriyel, aviyonik ve askeri) kullanılırlar

MINIX3: Modülerlik fikrinin en uç noktası, işletim sisteminin büyük bölümünü bağımsız kullanıcı süreçleri

POSIX uyumlu, açık kaynak kodlu bir sistemdir

Intel CPU'larındaki yönetim motoru

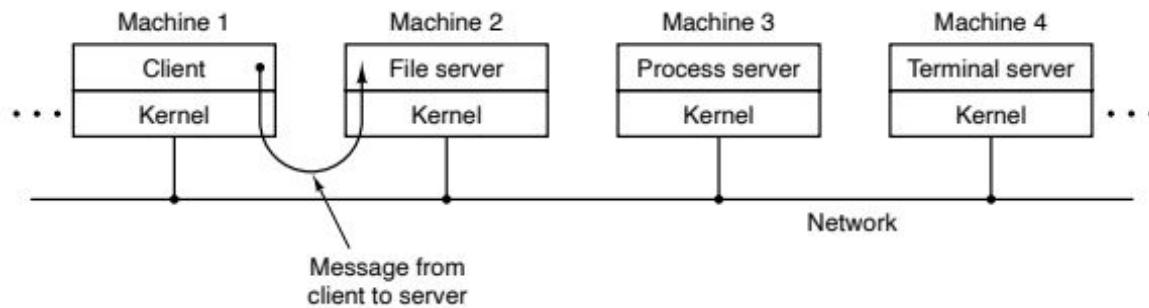


İstemci Sunucu Modeli

Süreçler: Sunucu ve istemci

İstemciler ve sunucular arasındaki iletişim genellikle mesaj geçişi yoluyla gerçekleşir.

Giderek artan sayıda sisteme (kullanıcılar bilgisayarlarda) istemci olarak, başka yerlerdeki büyük makineler ise sunucu olarak çalışmaktadır



Sanal Makineler

Sanallaştırma, internet sayfası sağlayıcıları arasında oldukça popülerdir.

Paylaşımlı (Sunucu üzerinde oturum açma dışında başka yetkilendirme yok.)

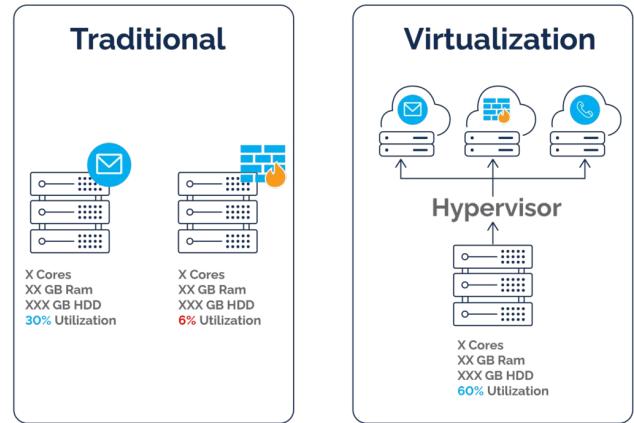
Özel (makinenin tüm yetkisi verilir)

Hizmet sağlayıcı: Tek bir makinede birden fazla sanal makine çalışma;

Hizmet alan: Kendilerine tahsis edilen sanal makine üzerinde tüm yetkiye sahiptirler

Son kullanıcı:

Windows kullanıcı: Sanal Linux makinesi kurmak



Sanal Makineler

Type 1 Hypervisor (Sanal Makine Monitörü):

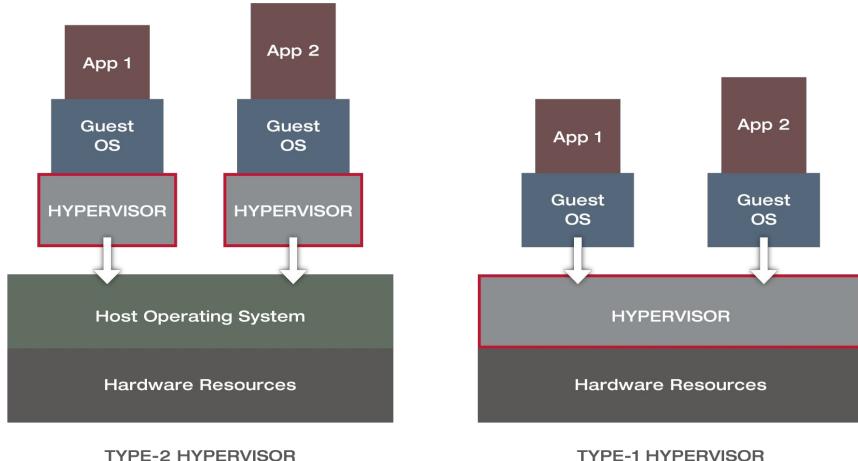
İki veya daha fazla işletim sisteminin aynı anda çalıştırılmak

Stanford ve Cambridge Üniversitelerinde araştırma makaleleri(1990) birkaç ticari ürünün (örneğin VMware Workstation ve Xen) ortayamasına ve sanal makinelere olan ilginin yeniden canlanmasına yol açmıştır.

Günümüzde popüler hipervizörler arasında KVM (Linux çekirdeği için), VirtualBox (Oracle tarafından) ve Hyper-V (Microsoft tarafından) yer almaktadır.

Kod bloklarını dahili bir önbellekte saklayıp yeniden kullanarak Bochs gibi yorumlayıcılara göre performansı artırması makine simülatörleri olarak adlandırılacağımız şeye yol açtı. Bu teknik sorunları iyileştirmeye yardımcı olsa da, ortaya çıkan sistemler akademik konferanslarda makale yayımlamak için yeterince iyi olsa da, performansın çok önemli olduğu ticari ortamlarda kullanılacak kadar hızlı değildi.

Performansı artırmadan bir sonraki adımı, Şekil 1-29(c)'de gösterildiği gibi ağır işlerin bir kısmını yapmak için bir çekirdek modülü eklemekti. Şu anda teknoloji VMWARE ile birlikte işbirliği halinde, tüm hizmetlerin



Sanal Makineler

Type 1 Hypervisor (Sanal Makine Monitörü):

İki veya daha fazla işletim sisteminin aynı anda çalıştırılmak

Stanford ve Cambridge Üniversitelerinde araştırma makaleleri(1990) birkaç ticari ürünün (örneğin VMware Workstation ve Xen) ortaya çıkışına ve sanal makinelere olan ilginin yeniden canlanmasına yol açmıştır.

Günümüzde popüler hipervizörler arasında KVM (Linux çekirdeği için), VirtualBox (Oracle tarafından) ve Hyper-V (Microsoft tarafından) yer almaktadır.

Kod bloklarını dahili bir önbellekte saklayıp yeniden kullanarak yorumlayıcılara göre performansı artırması makine simülatör adlandıracagımız şeye yol açtı. Bu teknik sorunları iyileştirmeye yardımcı ortaya çıkan sistemler akademik konferanslarda makale yayımlamak için olsa da, performansın çok önemli olduğu ticari ortamlarda kullanılacak değildi.

Performansı artırmak bir sonraki adımı, Şekil 1-29(c)'de göstergelerin bir kısmını yapmak için bir çekirdek modülü eklemekti. Şu anda teknoloji VM üzerindeki işlevlerini tam olarak hizmet etmektedir.

