



# OPERACIJSKI SUSTAVI

Kernel OSa  
Boot-anje

# Što ćemo danas raditi?

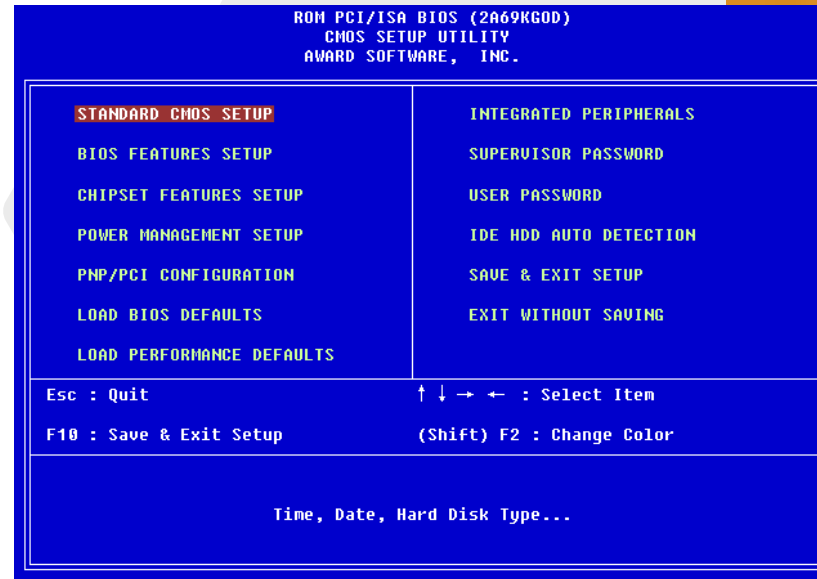
- Uloga BIOSa
- Kernel/jezgra Operativnog sustava
- Proces pokretanja Operativnog sustava – Boot
  - Windows
  - Linux

**Pokretanje Operativnog sustava**

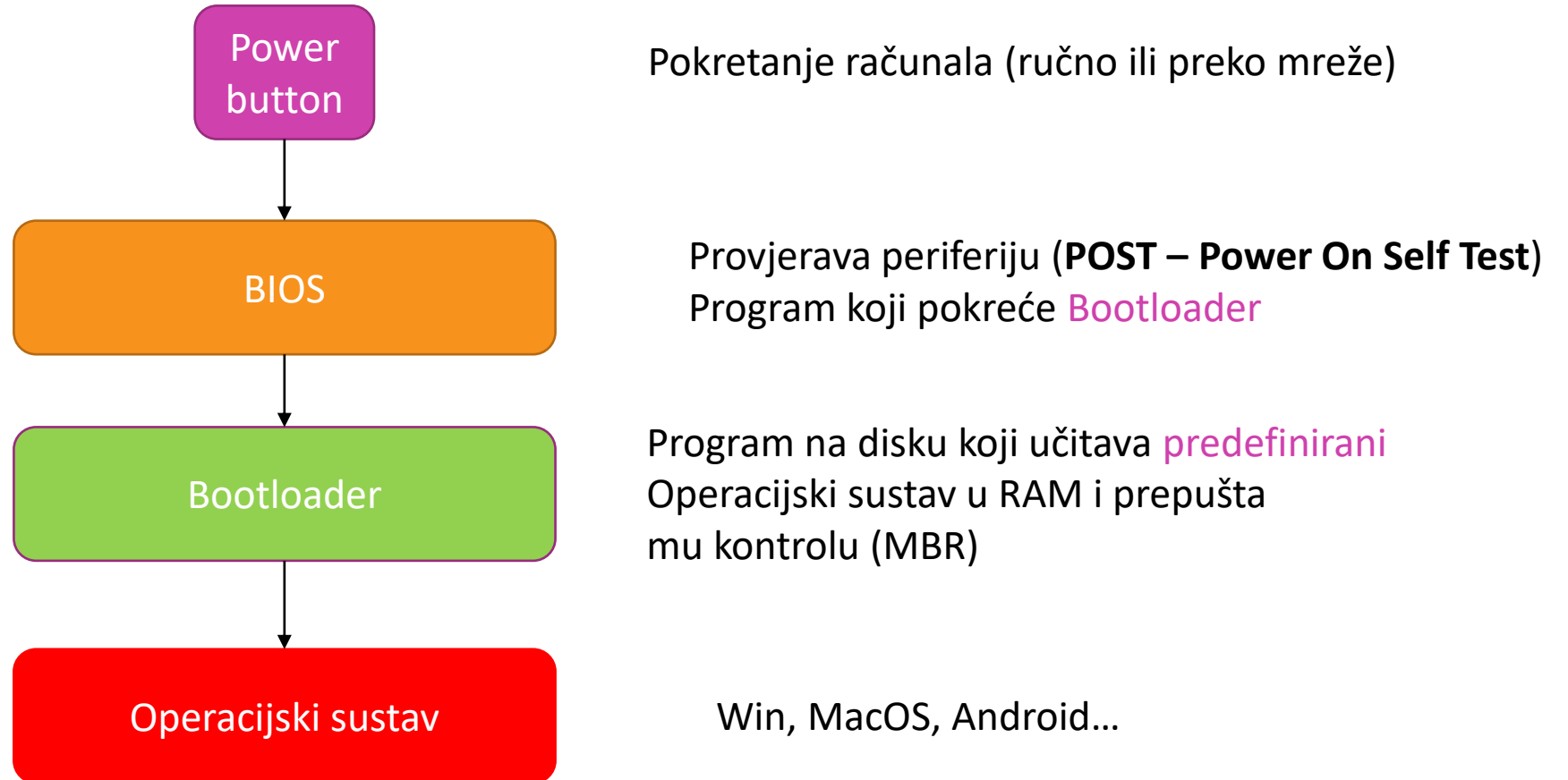
**BOOT(anje)**

# BIOS

- **B**asic **I**nput/**O**utput **S**ystem
- Računalni program (u binarnom obliku) koji se nalazi u EPROM čipu
- Kod pokretanja računala aplikacija koja se nalazi u BIOSu postaje proces u CPU i služi za pokretanje Operativnog sustava (koji se nalazi na disku)
- Prepoznaje koje su vanjske jedinice spojene na računalo (tastatura, miš, diskovi, grafička kartica...)



# Sekvenca pokretanja (boot) Operativnog sustava



<https://youtu.be/PSnGuvyIWBI>

# **Operacijski sustav**

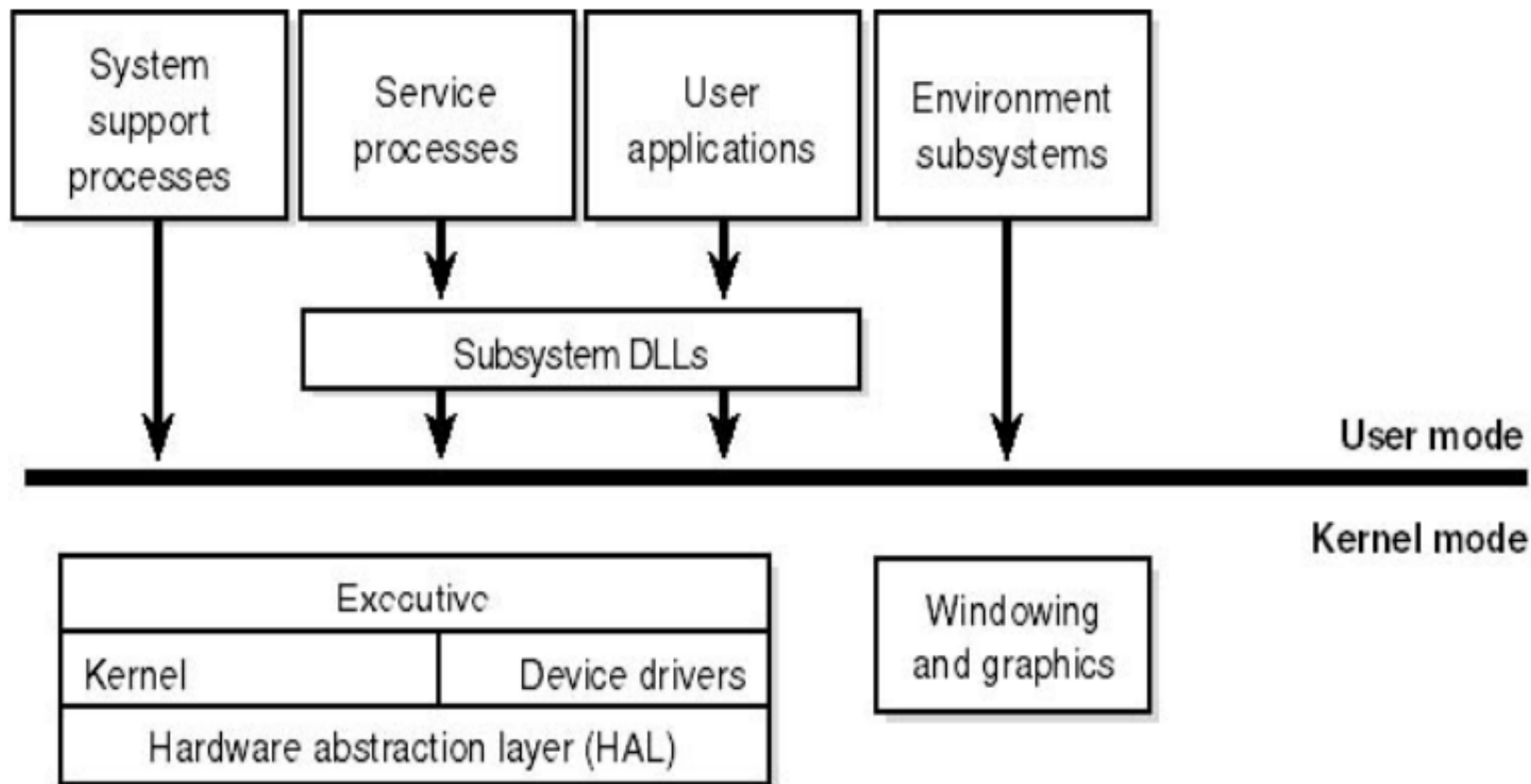
- Kernel**

- Prstenovi**

# Uloga kernela u Operativnom sustavu

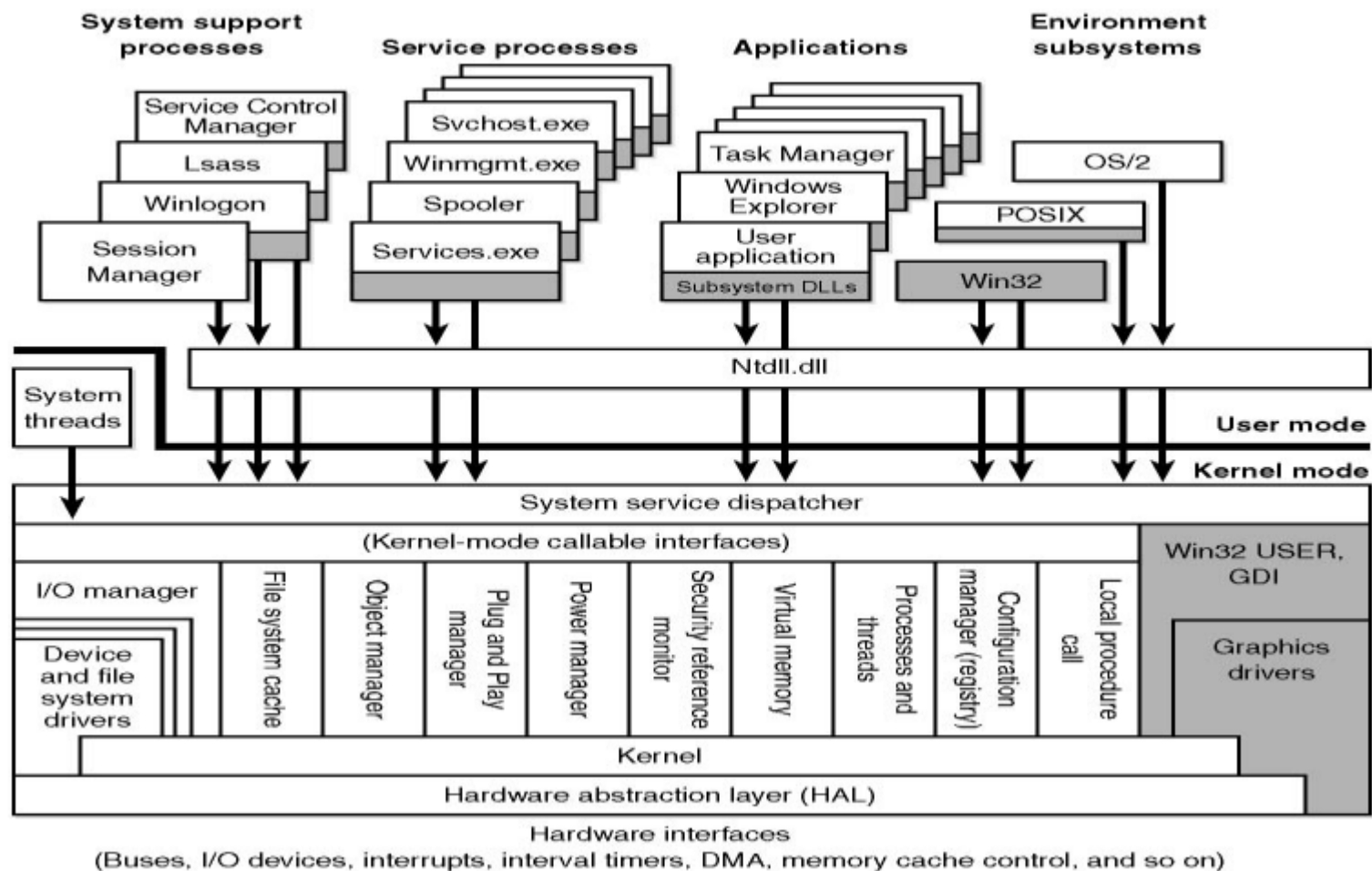
- **Upravljanje procesima**
  - Kreiranje i terminacija procesa
  - Raspored izvršavanja (*scheduling*)
  - Prebacivanje iz procesa u proces (*process switching*)
  - Sinkronizacija procesa i podrška za prekide
  - Upravljanje procesnim kontrolnim blokovima
- **Upravljanje memorijom**
  - Dodjeljivanje (alokacija) adresnog prostora
  - Zamjena (*Swapping*) memorijskog prostora
  - „Straničenje” (*Page and segment management*)
- **Upravljanje sa Ulazno/Izlaznim jedinicama**
  - Upravljanje međuspremnikom (*buffer*)
  - Dodjela I/O kanala i uređaja procesima
- **Funkcijske podrške**
  - Upravljanje prekidima
  - Nadzor i praćenje procesa

# Pojednostavljeni prikaz arhitekture

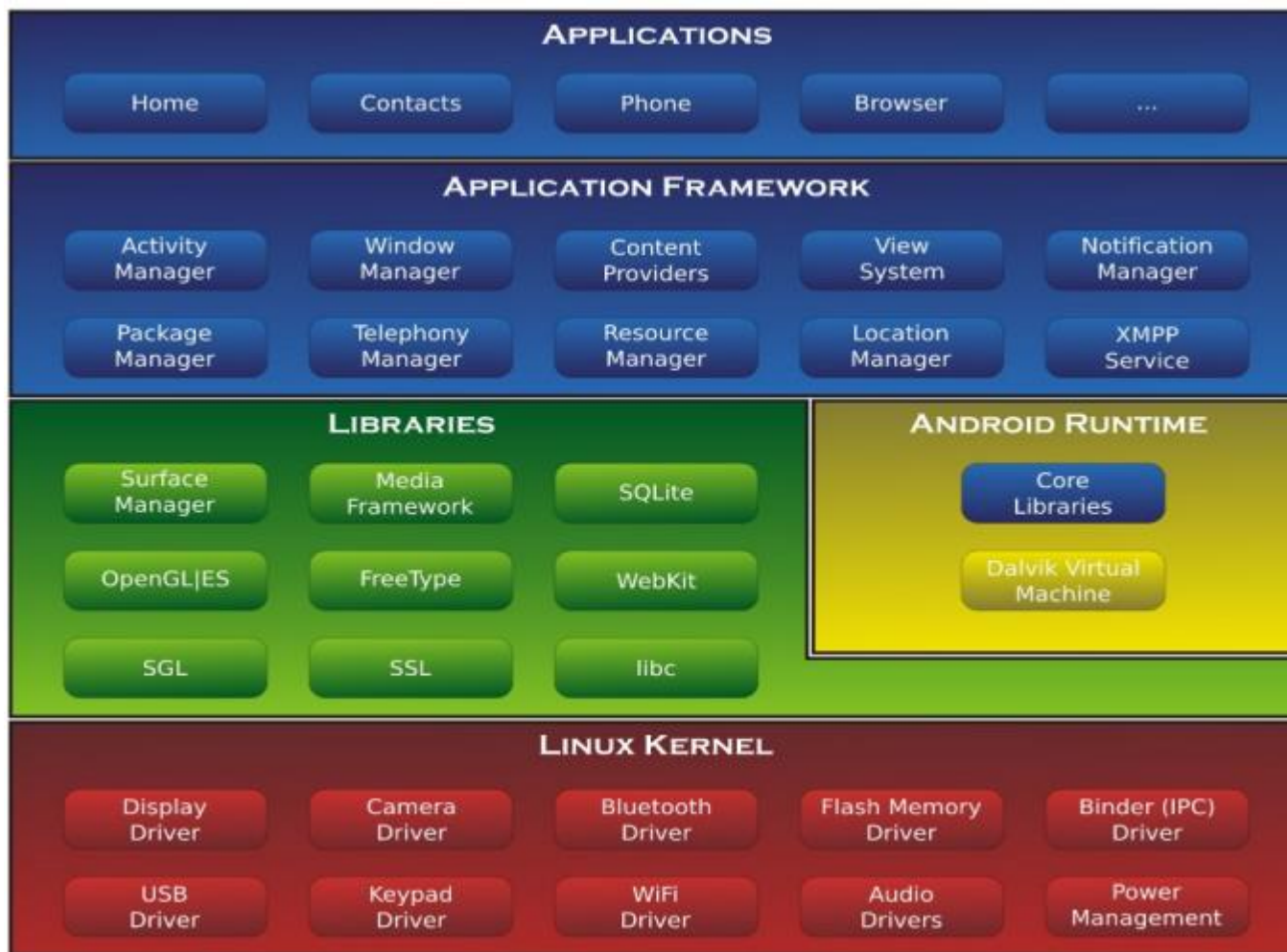




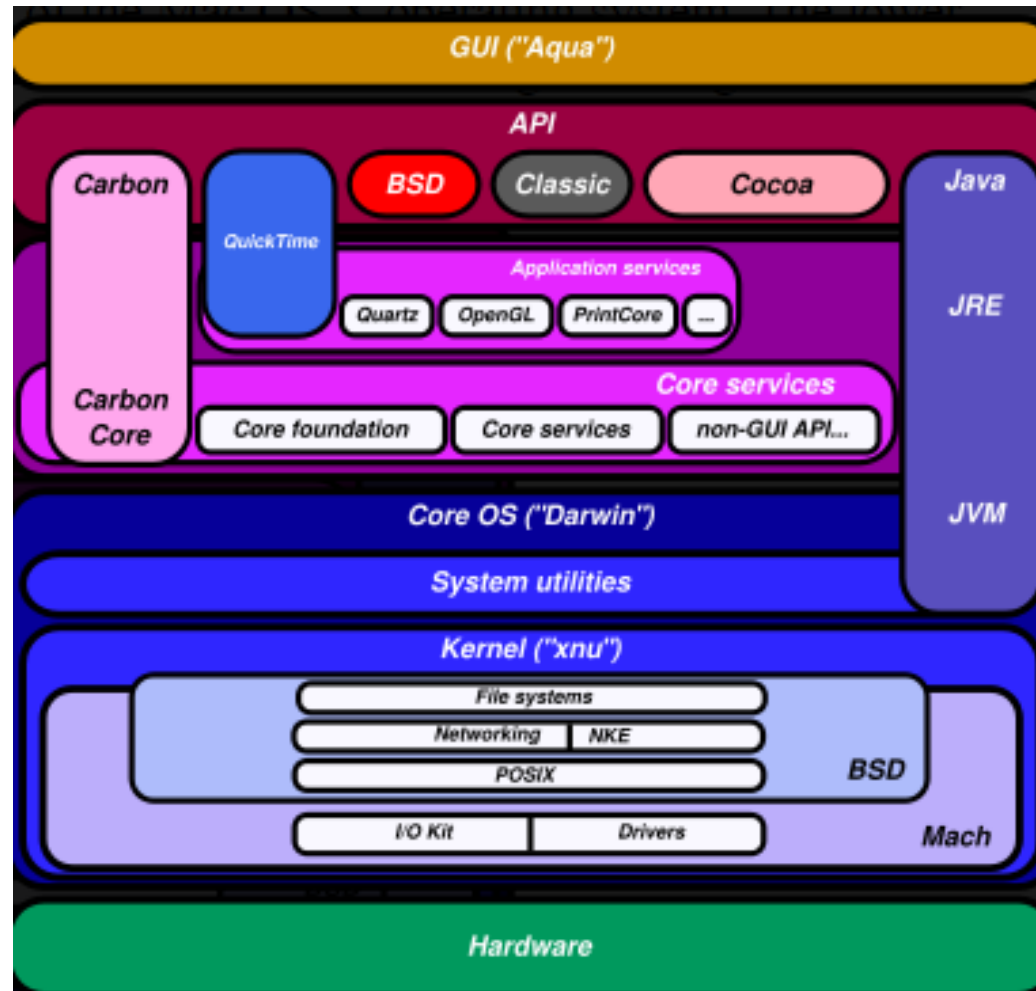
# Detaljni dijagram Windowsa



# Detaljni dijagram Androida



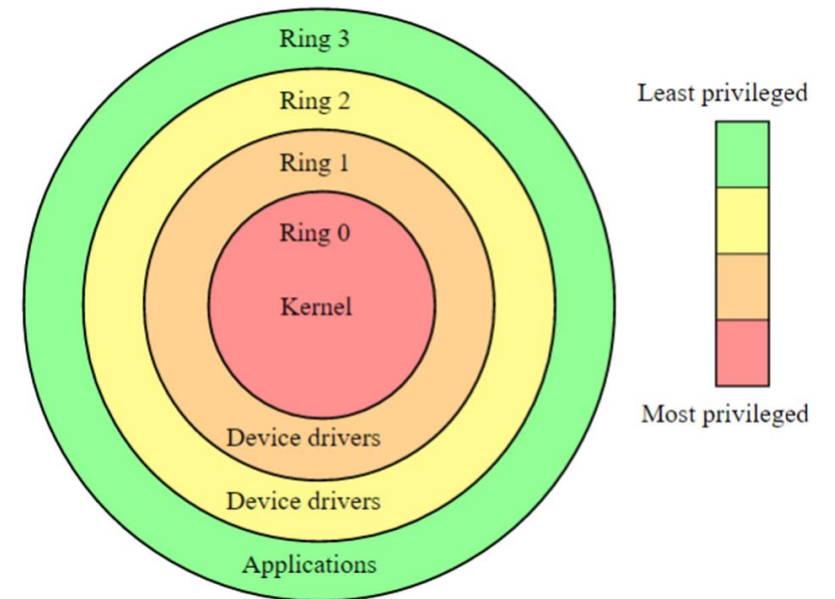
# Diagram Mac OSX



**Prstenovi**

# Prstenovi u procesoru

- Većina današnjih OS-ova koristi dva prstena
  - Prsten 0 – Kernel mod
  - Prsten 3 – User mod
- MS DOS je cijelo vrijeme bio u prstenu 0
- Prsteni 1 i 2 se rijetko koriste
- Prebacivanje iz jednog u drugi prsten je skupa radnja
  - 1000 – 1500 ciklusa
  - 70 iz jezgrinog u korisnički mod rada i 40 nazad dok je ostatak „kernel overhead”



# Korisnički mod vs. jezgrin mod (#1)

- Windowsi koriste dva načina rada (pristupa) procesora, radi zaštite kritičnih podataka:
  - Korisnički mod (User mod)
  - Jezgrin mod (Kernel mod)
- Korisničke aplikacije se izvršavaju u korisničkom modu, a kôd OS-a (npr. sistemski servisi i upravljački programi) u jezgrinom modu
  - Jezgrin mod se odnosi na način rada procesora koji dopušta pristup cjelokupnoj memoriji i svim CPU instrukcijama (prsten 0)
  - Napomena: hardver podržava 4 prstena, ali se najčešće koriste samo 2

# Korisnički mod vs. jezgrin mod (#2)

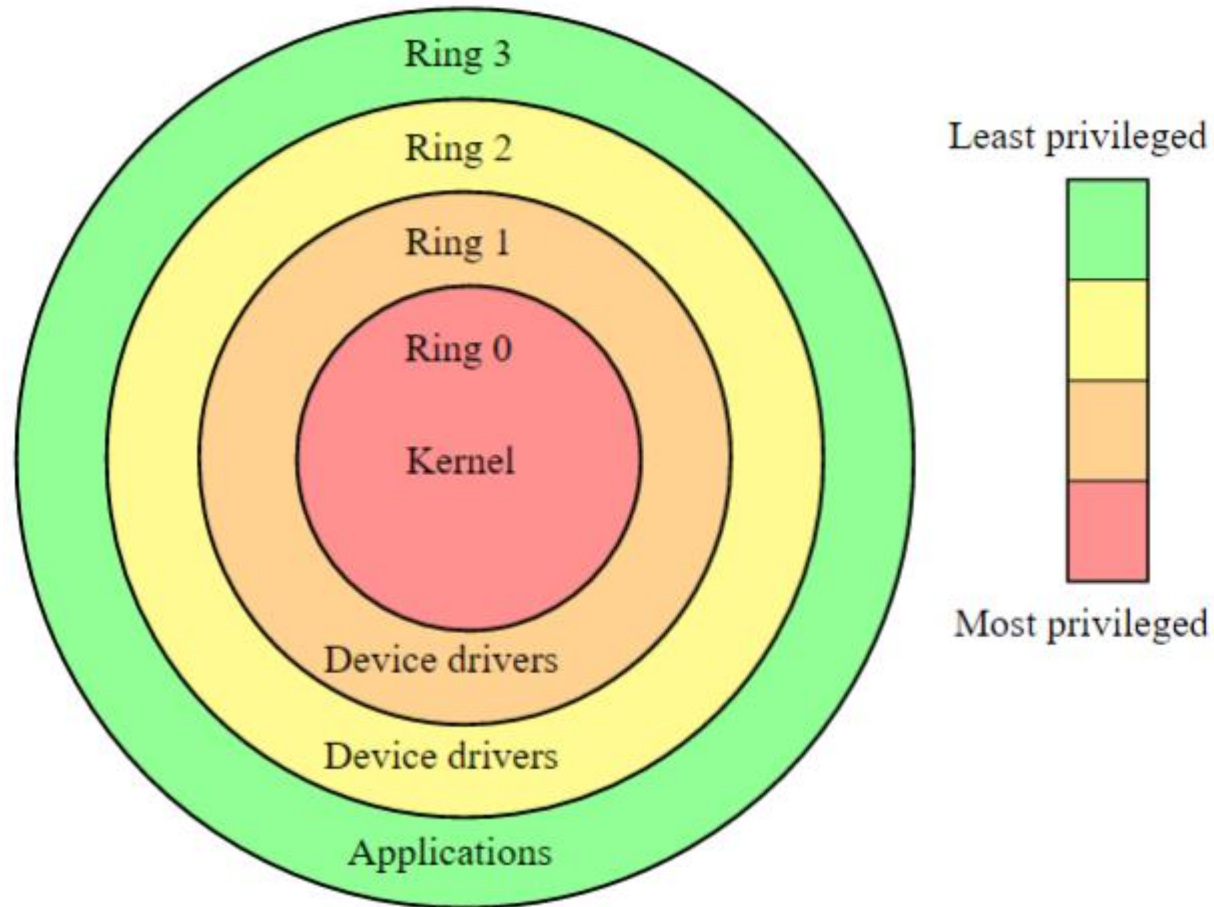
- Aplikacije se prebacuju u jezgrin mod prilikom poziva sistemskog servisa
- Prijelaz u jezgrin mod odrađuju specijalne CPU instrukcije
- Normalno je da korisnička dretva dio vremena izvršavanja provede u korisničkom, a dio u jezgrinom modu
  - Npr. Win32 ReadFile funkcija na kraju pozove internu Windows rutinu koja zaista čita datoteku.
  - S obzirom da rutina pristupa internom dijelu datotečnog sustava, mora se izvršiti u jezgrinom modu

# Prstenovi u procesoru

- Služe zaštiti
  - Npr. maliciozni softver ne može upaliti kameru bez da mu to korisnik dozvoli
- Prsteni su uvedeni s operacijskim sustavom MULTICS (1965)
- Većina današnjih procesora podržava 4 prstena
  - (napomena) ARM procesor nema hijerarhiju prstena
- Podržavaju različite modove rada procesora



# Grafički prikaz prstenova

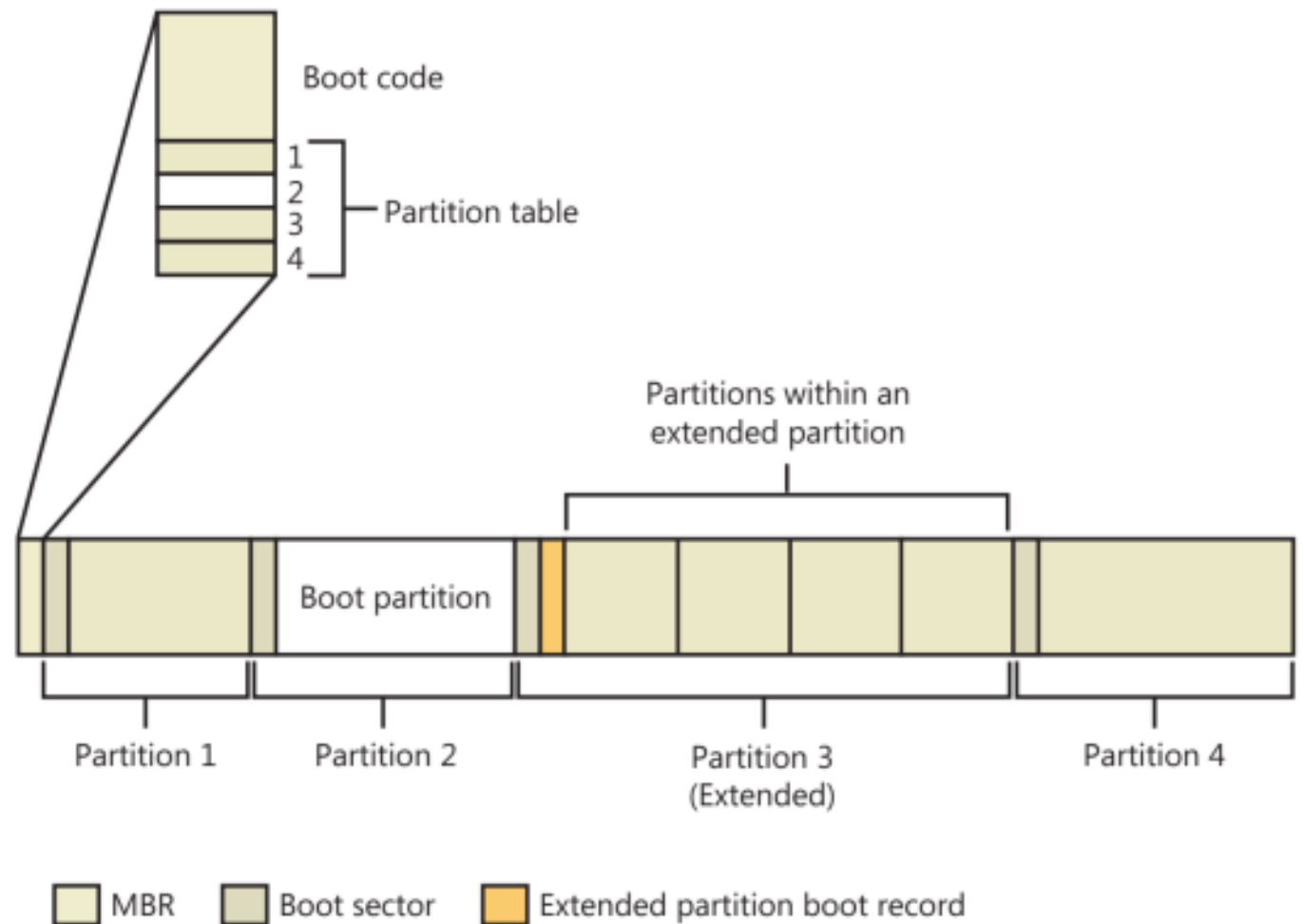


**WINDOWS**



# Windows preboot

- Boot proces počinje nakon instalacije Windowsa
- Windows Setup zapisuje MBR (Master Boot Record) i boot sektor na bootabilnu particiju:
  - Formatira particiju
  - Kreira `boot.ini` datoteku



# Boot.ini - legacy

- Popisuje operacijske sustave
- Sadrži boot parameter

```
[boot loader]
```

```
timeout=30
```

```
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS  
[operating systems]
```

```
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Microsoft  
Windows XP Professional" /fastdetect
```

```
C:\="Microsoft Windows XP"
```

# BCDEdit

- Binarna inačica boot.ini datoteke
- BCDEdit je komandno-linijski alat za upravljanje konfiguracijom podizanja sustava (Boot Configuration Data - BCD)
- BCD dokument pohranjuje informacije koje opisuju moguće
- boot procedure i njihove postavke

# BCDEdit

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>BCDEdit

Windows Boot Manager
-----
identifier                {bootmgr}
device                    partition=\Device\HarddiskVolume1
description                Windows Boot Manager
locale                    en-US
inherit                    {globalsettings}
default                    {current}
displayorder                {current}
toolsdisplayorder          {memdiag}
timeout                    30

Windows Boot Loader
-----
identifier                {current}
device                    partition=C:
path                      \WINDOWS\system32\winload.exe
description                Windows 10
locale                    en-US
```

# Pokretanje Windows operativnog sustava

## 1. BIOS

### 1. BIOS (u EPROMu) izvršava radnje:

- 1) Odabire boot uređaj i čita MBR (master boot record) kod
- 2) MBR locira boot particiju i prenosi kontrolu boot sektoru
- 3) Boot sektor učitava Ntldr u memoriju i izvršava ga

### 2. Ntldr se izvršava u 16-bitnom modu

- 1) Prebacuje sustav u zaštićen (protected) mod i uključuje straničenje (paging)
- 2) Koristi ili BIOS INT 13 ili Ntbootdd.sys driver za pristup disku, kako bi učitao **Boot.ini** datoteku
- 3) Ako postoji datoteka Hiberfil.sys u rootu particije, učitava ju u RAM te poziva kernel da pokrene sustav iz **hibernacije**

# Pokretanje Windows operativnog sustava

## 2. Ntldr

- 4) Ntldr prikazuje boot izbornik (ukoliko je potrebno)
  - 5) Izvršava **Ntdetect.com** datoteku koja popisuje hardversku konfiguraciju (pohranjuje se u HKLM\HARDWARE\DESCRIPTION)
  - 6) Ntldr se prebacuje u long mode ukoliko je OS 64-bitni
  - 7) Učitava **Ntoskrnl.exe** i **Hal.dll**
  - 8) Učitava sve boot upravljačke programe (specificirano u \Windows\System32\Config\System)
3. **Ntlrd** prepušta kontrolu učitanoj **kernelu** Windows OSa



# Pokretanje Windows operativnog sustava

## Faza 0

1. Faza 0 počinje s onemogućenim prekidnim sustavom
2. Inicijaliziraju se procesori i ostatak sustava.
3. HAL priprema kontrolor prekida
4. Upravitelj memorije kreira raspon i omogućuje pristup FS-u
5. Upravitelj objekata kreira tablicu handleova
6. Sigurnosni monitor priprema prvi pristupni token (namijenjen računu LocalSystem)
7. Upravitelj procesa kreira Idle i System procese, ali i dretvu Phase1 Initialization (neće biti pokrenuta prije omogućavanja prekidnog sustava)
8. Dretva Idle pokreće sljedeću fazu (phase 1)
9. HAL omogućuje prekidni sustav

# Pokretanje Windows operativnog sustava

## Faza 1

1. Boot video upravljački program (Bootvid.dll) prikazuje početnu animaciju
2. Inicijaliziraju se upravitelji potrošnje, vremena i procesora
3. Kreiraju se izvršni objekti i tablica servisa
4. Kreiraju se MM dretve
5. Ntdll.dll i NLS tablice se mapiraju u sistemski adresni prostor
6. Inicijalizira se upravitelj cache memorije
7. Upravitelj konfiguracije kreira registry bazu
8. Inicijalizira se PnP upravitelj
9. I/O upravitelj inicijalizira upravljačke programe
10. Kreira se Smss (Session Manager SubSystem) proces (sustav se ruši ako se ovaj proces završi za manje od 5 sekundi)
11. Dretva Phase1Initialization postaje zero page dretva

# Pokretanje Windows operativnog sustava

## Korisnički (User) mod

### 4. Smss se izvršava:

1. Kreiraju se LPC portovi za upravljanje sesijom
2. Pokreću se programi navedeni u BootExecute ključu registryja (Autochk)
3. Mapiraju se poznati DLL-ovi
4. Kreiraju se datoteke straničenja (Engl. Paging)
5. Upravitelj konfiguracije završava s inicijalizacijom registryja
6. Kreiraju se varijable okruženja
7. Win32k.sys se učitava – početak korištenja GUI-ja
8. Csrss (Client/Server Runtime SubSystem) se pokreće
9. Winlogon se pokreće
10. Kreiraju se LPC portovi namijenjeni debugiranju
11. Smss čeka handleove Csrss-a i Winlogona

# Pokretanje Windows operativnog sustava

## WIN Logon

### 5. Winlogon se izvršava:

1. Kreira se desktop i sustav prozora
2. GINA (Graphical Identification and Authentication) prikazuje okvir za prijavu na sustav
3. SCM (Service Control Manager, services.exe) učitava servise i upravljačke programe predodređene za auto start
4. Lsass (Local Security Authentication Subsystem) se kreira
5. Prijavu na sustav kroz GINA-u provjerava Lsass
6. U registry segment HKCU se učitava korisnički profil

### 6. GINA pokreće Userinit.exe

1. Userinit izvršava korisničke i skripte računala
2. Userinit pokreće shell (Explorer.exe)
3. Network provideri se obavještavaju o prijavi (mapiraju diskove, printere...)
4. Boot je završen te SCM ažurira Last Known Good Configuration  
(HKLM\SYSTEM\Select\LastKnownGood)

# Bootmgr i Winload

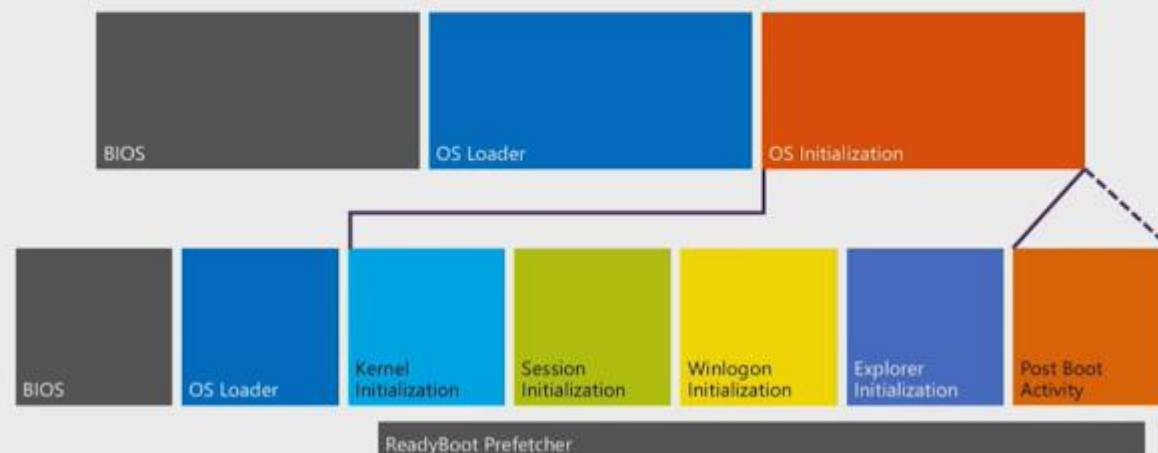
- Ntldr je zamijenjen sa:
  - **Bootmgr** čita BCD i prikazuje boot meni
  - **Winload** izvršava učitavanje OS-a
  - **Winresume** izvršava povrat iz hibernacije
- Podržani su i pre-boot izvršni alati, npr:
  - Memtest (Windows Memory Diagnostic)

# Pojednostavljeno (1)

Windows Boot: Processes Perspective



Windows Boot



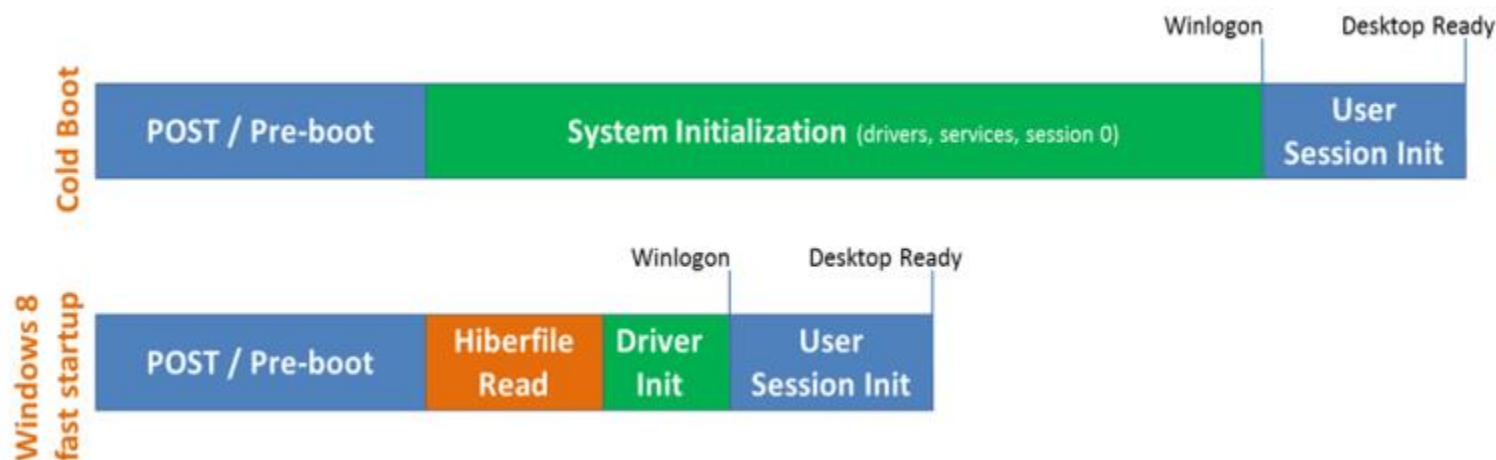
<https://windowsrunbook.blogspot.com/2016/02/windows-boot-process.html>

# Pojednostavljeno (2)

- Pročitaj MBR i prebaci se na boot sektor
- Pokreni Ntldr
- Prenesi kontrolu kernelu
- Inicijalizacija velikog broja komponenti
- Pokreni Smss
- Priprema za prijavu: Winlogon, Lsass, Csrss
- Nakon prijave: Userinit, shell

# Zašto se Win 8 i 10 brže podižu?

- Boot sistem je optimiziraniji da troši što manje hardverskih resursa
- Win8 prilikom gašenja gasi user sesiju, a kernel stavlja u hibernaciju
- Tradicionalno se gase obje sesije
- Najbolje radi na SSD tipovima diskova





# Zašto je 11 „brži” od 10

- <https://www.youtube.com/watch?v=olYHRRTCvy4&t=153s>
- Ušteda memorije i CPU do 35%
- 40% manji – pa se brže „boot-a”
- Fast Boot opcija
  
- *5 razloga ZA i 5 razloga protiv WIN11:*
  - <https://www.pcworld.com/article/1471886/why-switch-to-windows-11.html>



Windows 11

**Linux**



# Linux boot

BIOS obavlja radnje specifične za hardware

BIOS pokreće boot kôd s označenog uređaja

Pokreće se bootloader s opcijama boot-a

Start kernela

Init procesi ili SystemD

# Linux bootloaderi

- Dva najčešća
  - **LILO**
    - Agnostičan prema datotečnim sustavima
    - Koristi „sirove” podatke s diska s predefiniranim pozicijama diska
  - **GRUB**
    - Razumije ext2, ext3 i ext4
- Postoje još
  - SYSLINUX (Za boot s USB-a ili CD-a)
    - Razumije FAT, NTFS
  - Loadin (za boot iz DOS-a ili Win9x)

# Pokretanje Linux kernela

- Podizanje kernela u memoriju
  - Kernel se nalazi u image datoteci
- Pokretanje kernela
  - Inicijalizacija memorije i hardvera
  - Kompresiran initrd (initial ramdisk) image stavlja u memoriju i iz istog preuzima upravljačke programe
  - Inicijalizacija virtualnih datotečnih sustava
  - Uklanjanje intrid-a iz RAM-a
  - Kreiranje root uređaja i namještanje root particije u status čitanja (Engl. Read-only)
- Kernel je učitano, omogućeni su prekidi

### Boot PROM phase

- 1) BIOS runs POST
- 2) BIOS loads mboot

### Boot program phase

- 1) boot loads GRUB stage1
- 2) GRUB stage1 loads GRUB stage 2
- 3) GRUB stage 2 reads menu.lst
- 4) GRUB stage 2 loads primary boot archive and multiboot

### kernel initialization phase

- 1) kernel reads /etc/system
- 2) kernel initializes itself and loads modules

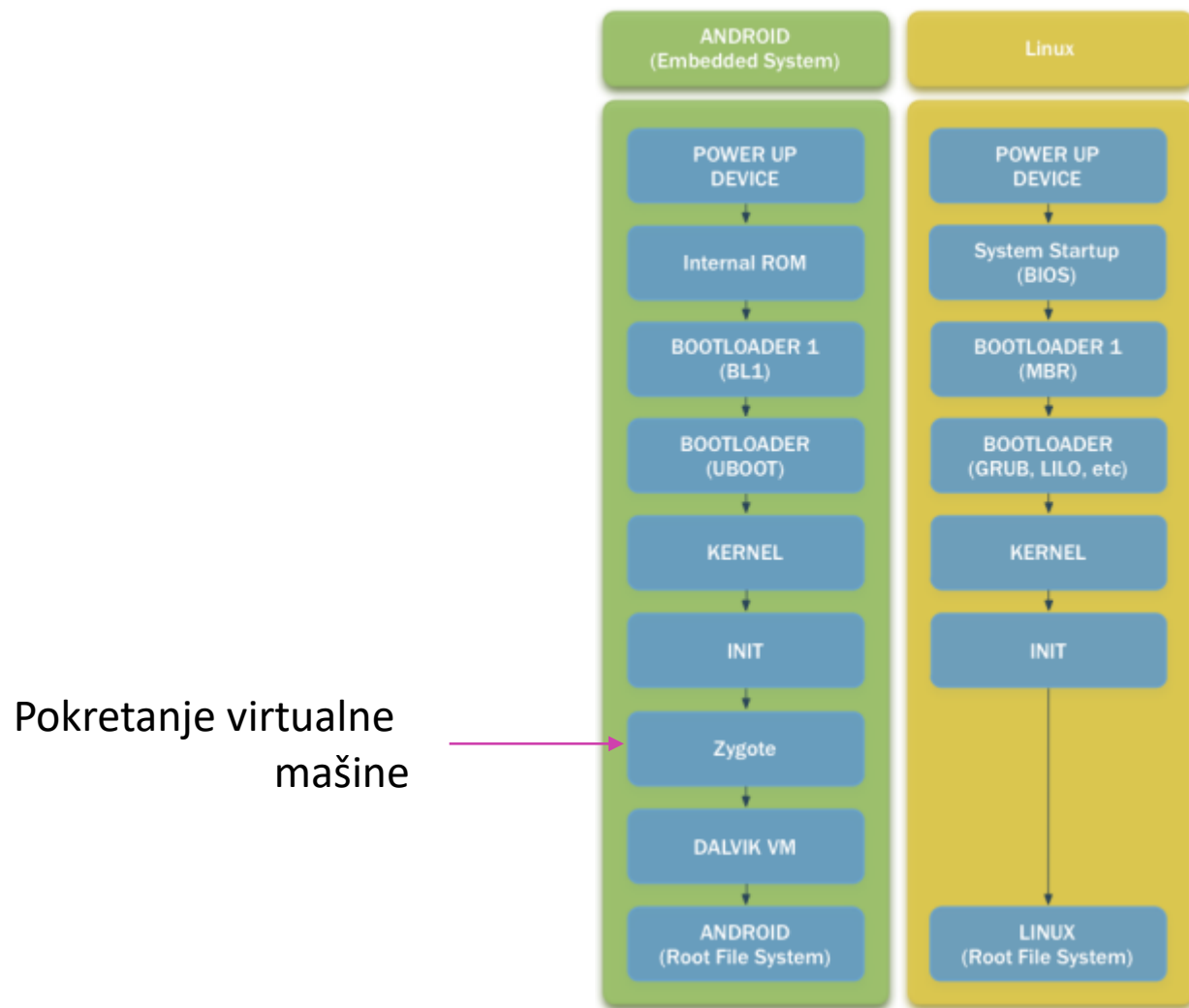
### init phase

- 1) kernel start /etc/init
- 2) /etc/init starts svc.startd process

### svc.startd phase

- 1) svc.startd starts the services

# Android vs. Linux





**Hvala na  
pažnji!**