

Sistemas Operacionais na prática

Bootloader

Roteiro

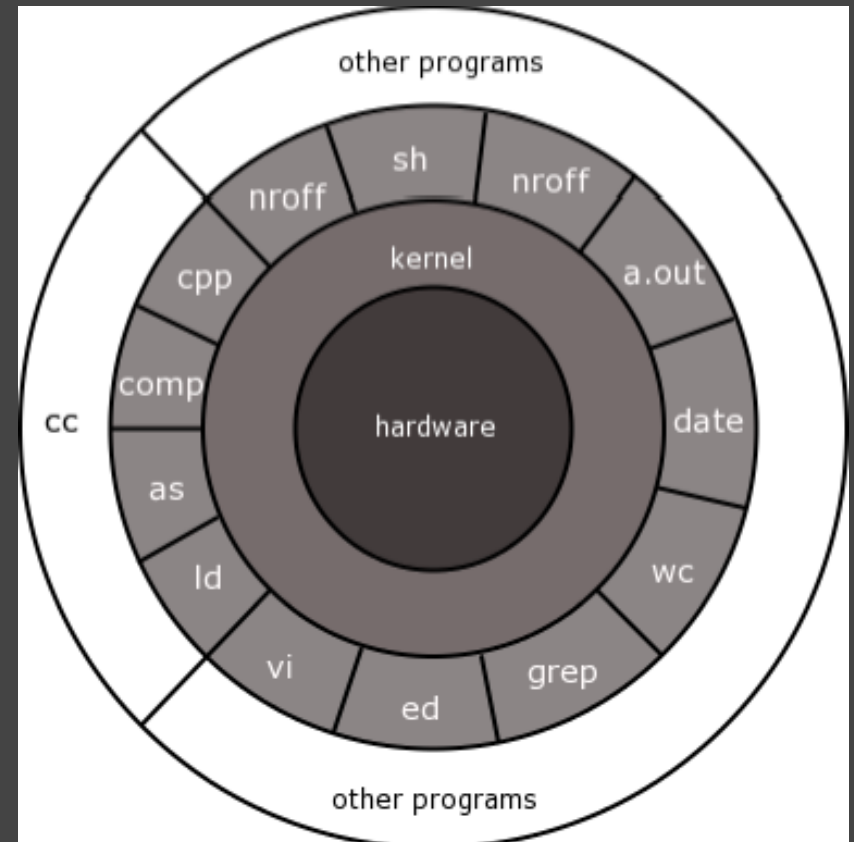
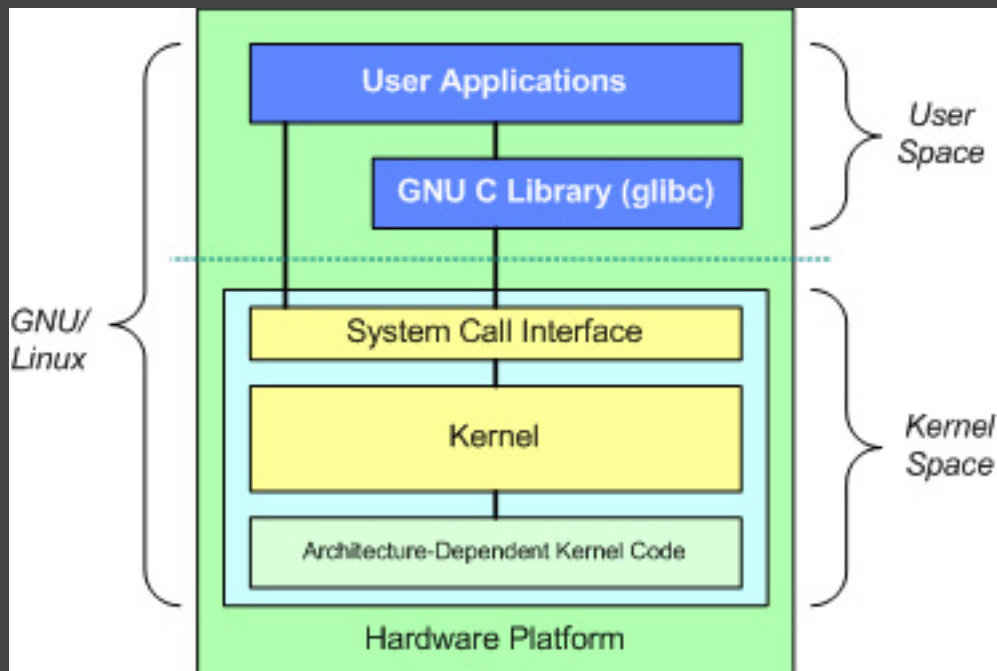
- Motivação
- Contexto
- Disco
- Processador
- Processo de Boot
 - BIOS
 - Bootloaders
- Projeto

Motivação

- O que acontece depois de ligar o PC?
- Como um SO é executado?
- Assunto BÁSICO.
- VAI CAIR NA PROVA.

Contexto de um SO

- Complexo
- Abstrai o hardware
- Gerencia recursos



Contexto de um SO

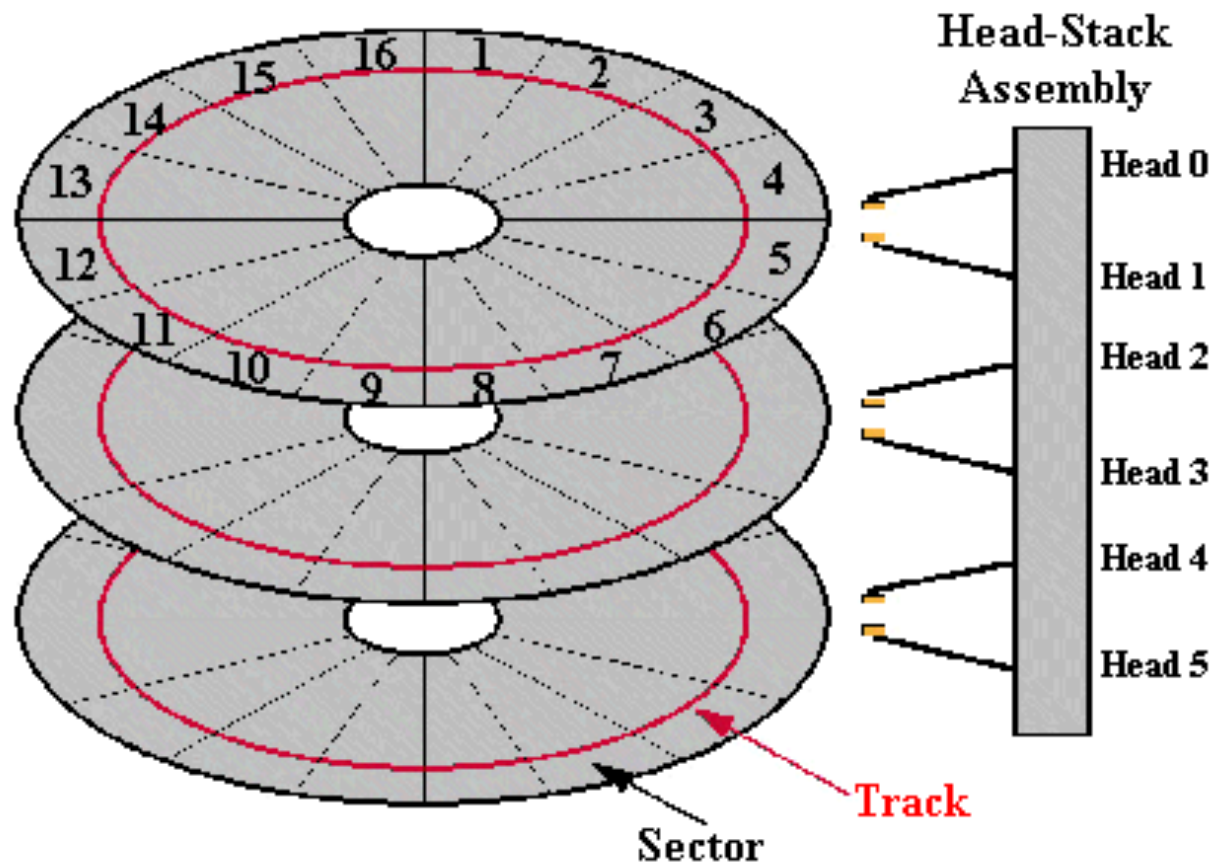
Problemas em Desenv. de um SO:

- Linguagem de Programação
- Bibliotecas
- Assembly
- Suporte à Dispositivos
- Não há mais ninguém pra culpar

Eu me garanto, como é que eu começo?

Disco Rígido

Drive Physical and Logical Organization



Disco Rígido



Disquete

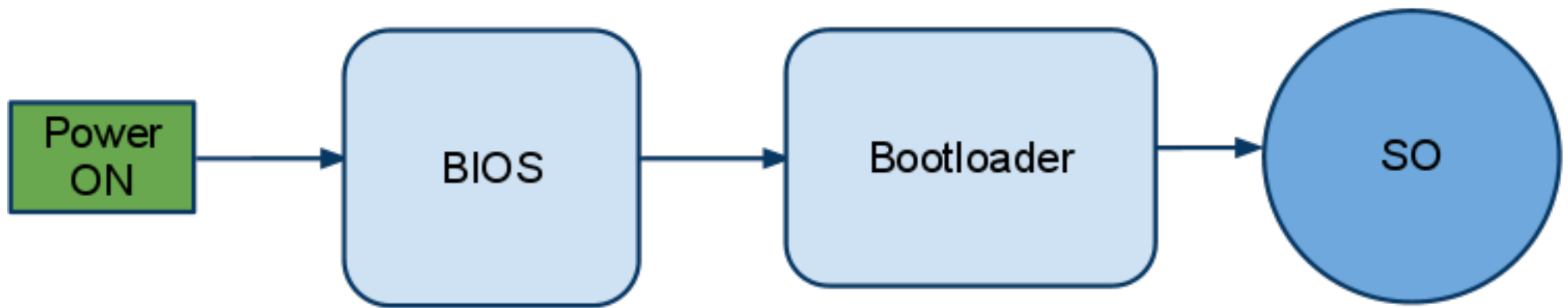
- Disquetes vs. Discos Rígidos
- CHS = Cylinder, Head, Sector
- Bootsector CHS => (0, 0, 1)
- Floppy 0 => drive 0
- Floppy => FAT12

Processador

- IP/PC (program counter) = ?
- Endereçamento Segmento:Offset (Modo Real)
- Modo Real (16 bits) vs. Modo Protegido (32 bits)
- 64 bits?

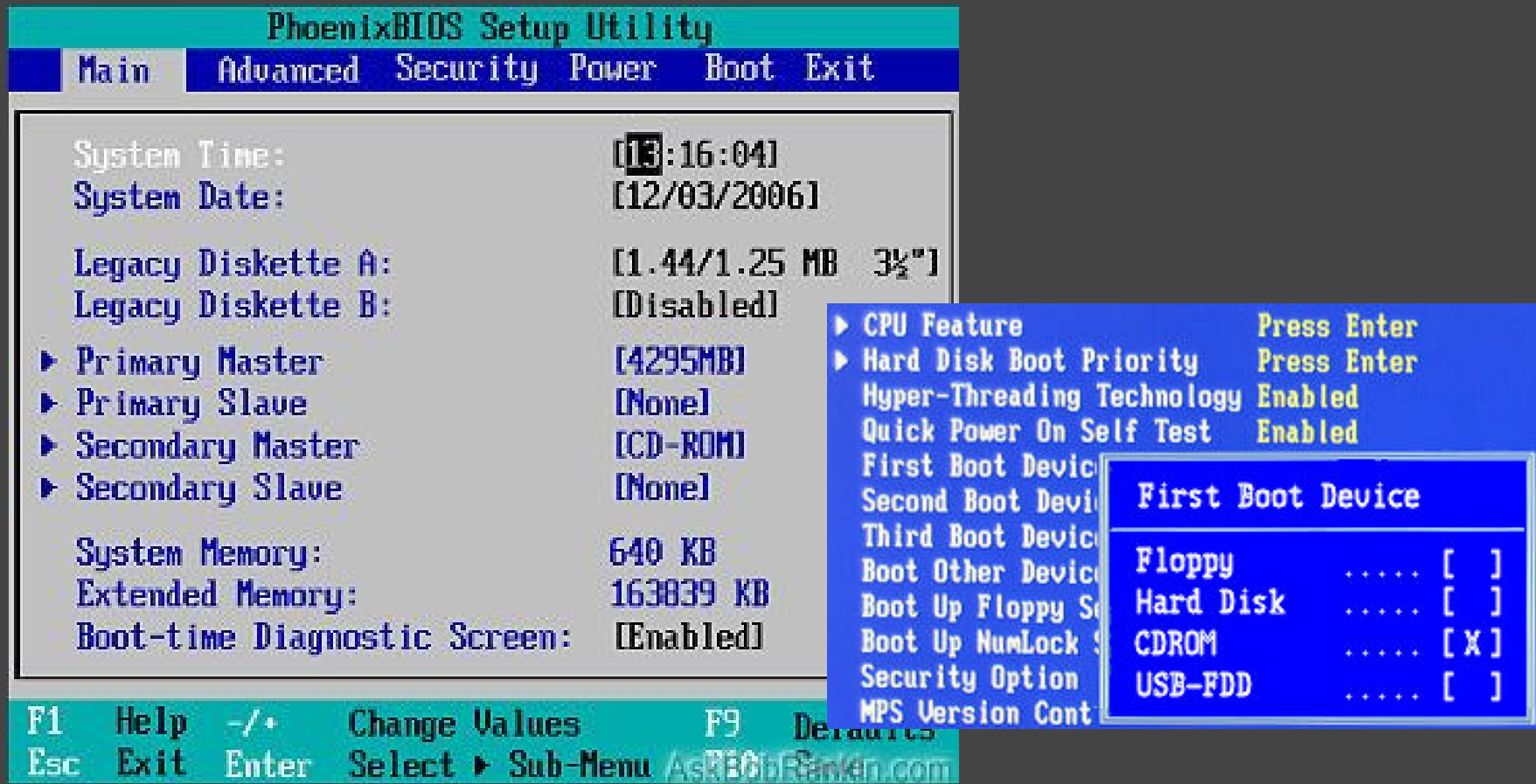
Processo de Boot

Processo de Boot
(simplificado)



BIOS (para leigos)

- Basic Input/Output System
- Ordem de boot
- Obter informações da máquina



BIOS

- Firmware
- Primeiro programa executado.
- "Partes":
 - POST (Power on Self Test)... BEEP
 - Menu (GUI)
 - Carrega o bootsector
 - Interrupções



BIOS

Interrupções

- Interrupt Vector Table (IVT) = IDT (x86)
- Interrupt Service Routine (ISR)

BIOS

Carrega o bootloader

- 0x7C00 (linear)
- 0x0000:0x7C00 vs. 0x07C0:0x0000
- CS = ?
- DS = ?
- DL = boot drive (floppy 0 = 0x0, hd 0 = 0x80)
- boot signature = 0xAA55 (little endian)

Bootloader

Objetivos

1. Carregar o kernel na memória.
2. Passar para o modo protegido (?).
3. Passar o controle para o kernel.

Bootloader

Quantos estágios?

- Bootloader de 1 estágio
- Bootloader de 2 estágios (ou mais)

Bootloader

Como carrega o kernel na memória?

- Carrega setores pré-determinados do disco
- Usa a formatação (sistema de arquivos) do disco para achar o kernel, ou seja, sabe o nome do arquivo no disco. (e.g. FAT 12)

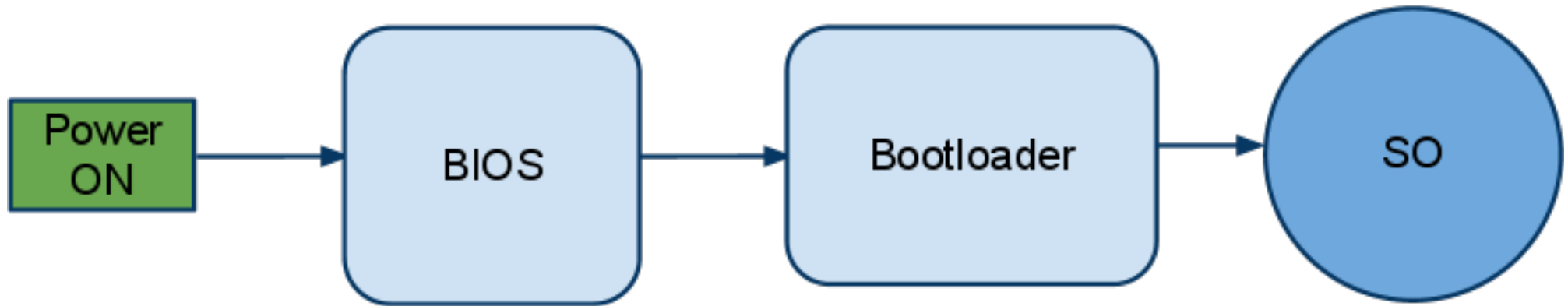
Bootloader

SO é 16 bits ou 32 bits?

- Permanece no modo real
- Passa para o modo protegido

Processo de Boot

Processo de Boot (simplificado)



- POST
- Bootloader => 0x7C00
- Pula para 0x7C00

- Inicializa estruturas
- Carrega kernel na memória
- Passa p/ modo protegido
- Pula para o kernel

Lab2 - Bootloader

Inicializando um SO

Objetivos Principais

- Entender o Processo de Boot
 - Saiba explicar todo o processo.
- Entender o código Assembly
 - Entenda o propósito de cada linha
- Entenda Bootloaders
 - Como eles carregam o SO
- Lab1 é pré-requisito :
 - Endereçamento Segment:Offset (essencial)

Ambiente de Desenv.

- Linux (Ubuntu)

As instruções do projeto foram feitas no Ubuntu, utilize outro ambiente por sua conta e risco.

- Nasm

O código base do bootloader, foi feito para o Nasm, utilizar outro assembler vai causar erros.

- Qemu

O Qemu foi usado como emulador de um x86.

Sugestões

- Instale o Linux em casa...
HOJE (pode ser numa maquina virtual, até mesmo no qemu)
- Não deixe pra véspera (bugs)
- Não copie de outros grupos! (perguntas)
- Não tente copiar da internet! (conheço a maioria)
- Realidade: mesmo se você copiar da internet, você ainda vai ter dificuldades de entender porque está funcionando.
- LEIA as referências e tudo que você encontrar.

Possíveis Questões (monitores)

- Como a BIOS detecta um disp. bootável?
- Quais os tipos de bootloaders e por que existem?
- O que acontece quando se chama uma interrup.?
- Explique o processo de boot.
- Disserte sobre os modos do pentium.
- Explique o uso e a razão da existência dos registradores de segmento.
- Se você tirar ESSA linha de código...o que acontece? por que?

Referências

<http://wiki.osdev.org/>

<http://www.osdever.net/>

<http://www.brokenthorn.com/Resources/OSDevIndex.html>

<http://www.supernovah.com/Tutorials/>