

Práctica de Organización del Computador II

System Programming

Primer Cuatrimestre 2022

Organización del Computador II

DC - UBA

Pasaje a modo protegido

En esta parte vamos a ver:

En esta parte vamos a ver:

- Bootloader

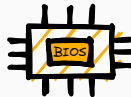
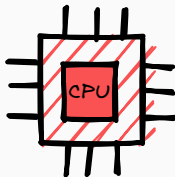
En esta parte vamos a ver:

- Bootloader
- Armado de GDT

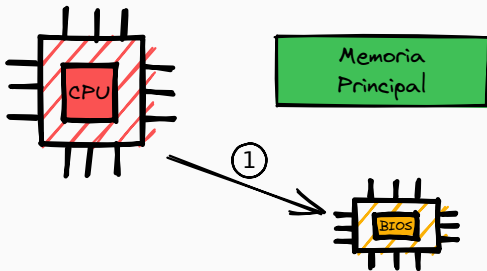
En esta parte vamos a ver:

- Bootloader
- Armado de GDT
- Pasaje a modo protegido

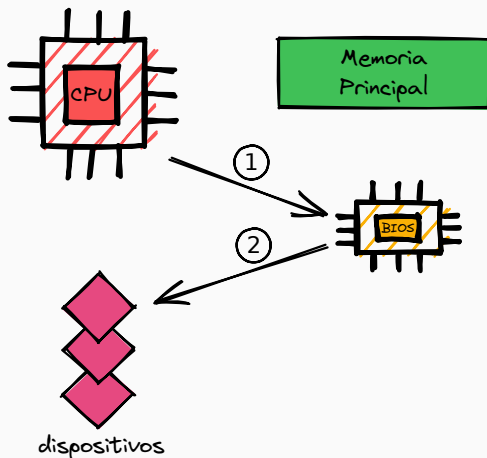
Bootloader



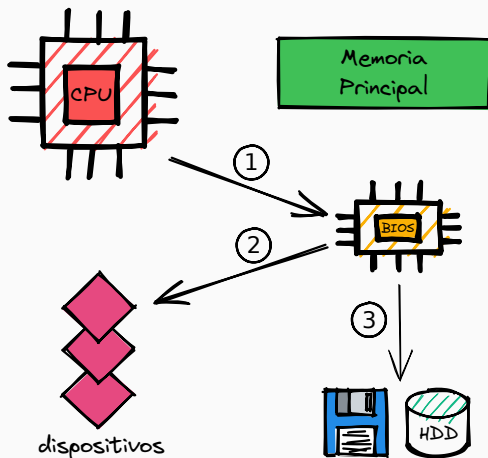
1. CPU ejecuta código residente en memoria flash de BIOS



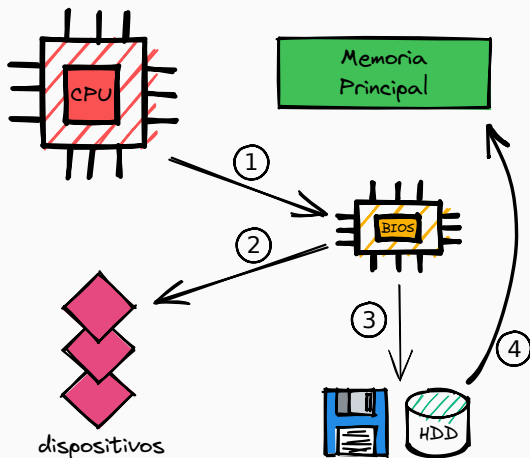
1. CPU ejecuta código residente en memoria flash de BIOS
2. BIOS ejecuta POST (Power On Self Test) en los dispositivos

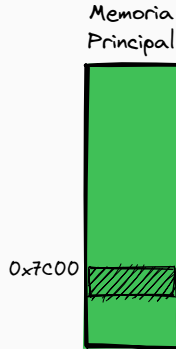


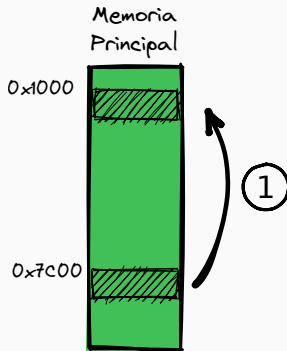
1. CPU ejecuta código residente en memoria flash de BIOS
2. BIOS ejecuta POST (Power On Self Test) en los dispositivos
3. BIOS busca un dispositivo "bootable"



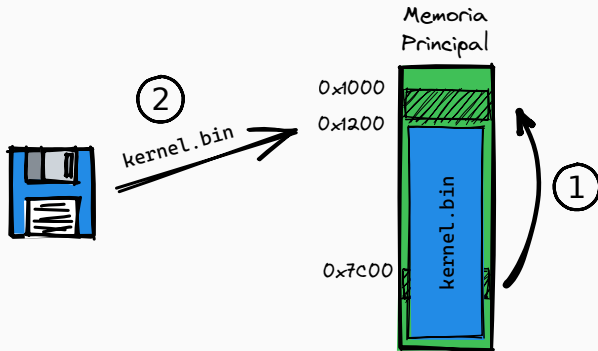
1. CPU ejecuta código residente en memoria flash de BIOS
2. BIOS ejecuta POST (Power On Self Test) en los dispositivos
3. BIOS busca un dispositivo "bootable"
4. Se copia a memoria principal en la posición 0x7C00 el sector de boot (512 bytes)



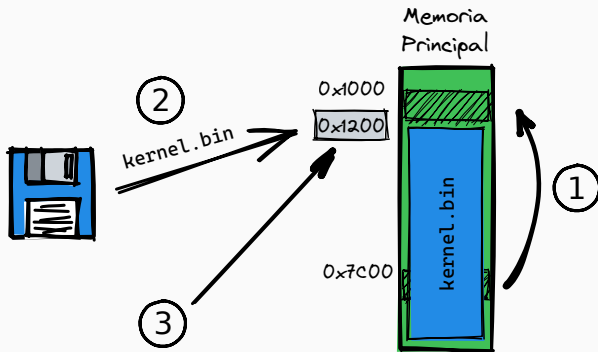




1. Se copia el bootloader a la posición **0x1000**



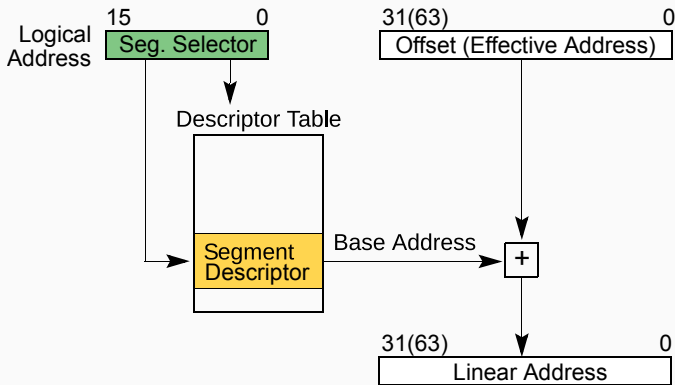
1. Se copia el bootloader a la posición **0x1000**
2. Busca y carga el archivo `kernel.bin` contenido en el diskette y lo copia en la dirección **0x1200**

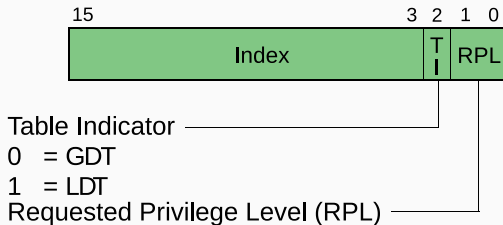


1. Se copia el bootloader a la posición **0x1000**
2. Busca y carga el archivo `kernel.bin` contenido en el diskette y lo copia en la dirección **0x1200**
3. Se salta hacia la dirección **0x1200** y se ejecuta desde ahí

Armado de GDT

Antes de hablar de la GDT, repasemos segmentación:





CS: Para acceder a código

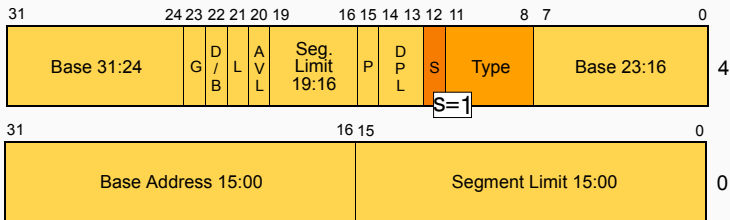
SS: Para acceder a pila

DS: Para acceder a datos (default)

ES: Para acceder a datos

GS: Para acceder a datos

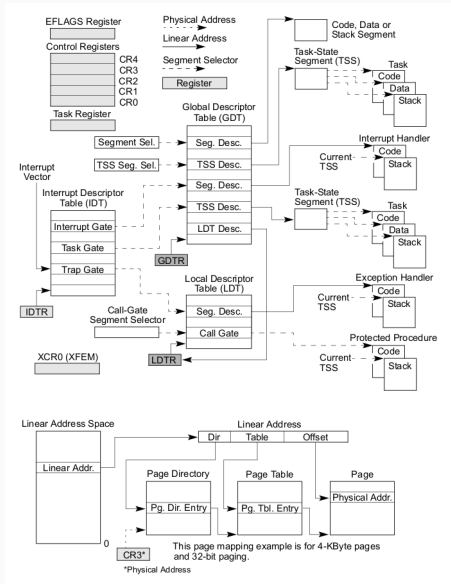
FS: Para acceder a datos

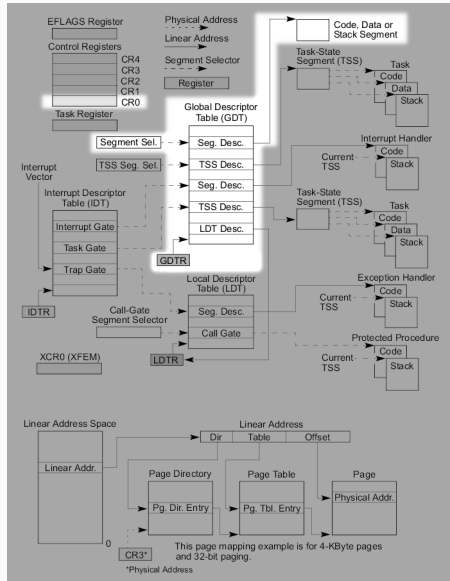


- L — 64-bit code segment (IA-32e mode only)
- AVL — Available for use by system software
- BASE — Segment base address
- D/B — Default operation size (0 = 16-bit segment; 1 = 32-bit segment)
- DPL — Descriptor privilege level
- G — Granularity
- LIMIT — Segment Limit
- P — Segment present
- S — Descriptor type (0 = system; 1 = code or data)
- TYPE — Segment type

Type

Decimal	Type Field				Descriptor Type	Description
	11	10 E	9 W	8 A		
0	0	0	0	0	Data	Read-Only
1	0	0	0	1	Data	Read-Only, accessed
2	0	0	1	0	Data	Read/Write
3	0	0	1	1	Data	Read/Write, accessed
4	0	1	0	0	Data	Read-Only, expand-down
5	0	1	0	1	Data	Read-Only, expand-down, accessed
6	0	1	1	0	Data	Read/Write, expand-down
7	0	1	1	1	Data	Read/Write, expand-down, accessed
		C	R	A		
8	1	0	0	0	Code	Execute-Only
9	1	0	0	1	Code	Execute-Only, accessed
10	1	0	1	0	Code	Execute/Read
11	1	0	1	1	Code	Execute/Read, accessed
12	1	1	0	0	Code	Execute-Only, conforming
13	1	1	0	1	Code	Execute-Only, conforming, accessed
14	1	1	1	0	Code	Execute/Read, conforming
15	1	1	1	1	Code	Execute/Read, conforming, accessed





- Completar la GDT

- Completar la GDT
- Deshabilitar interrupciones

- Completar la GDT
- Deshabilitar interrupciones
- Cargar el registro GDTR con la dirección base de la GDT

- Completar la GDT
- Deshabilitar interrupciones
- Cargar el registro GDTR con la dirección base de la GDT
- Setear el bit PE del registro CR0

- Completar la GDT
- Deshabilitar interrupciones
- Cargar el registro GDTR con la dirección base de la GDT
- Setear el bit PE del registro CR0
- FAR JUMP a la siguiente instrucción
 `JMP <selector>:<offset>`

- Completar la GDT
- Deshabilitar interrupciones
- Cargar el registro GDTR con la dirección base de la GDT
- Setear el bit PE del registro CR0
- FAR JUMP a la siguiente instrucción
 `JMP <selector>:<offset>`
- Cargar los registros de segmento (DS, ES, GS, FS y SS)

