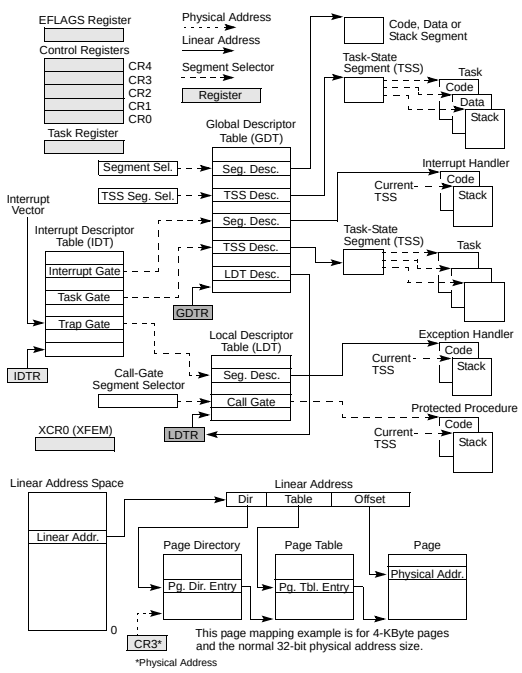
Paginación Identity Mapping

Programación de Sistemas Operativos

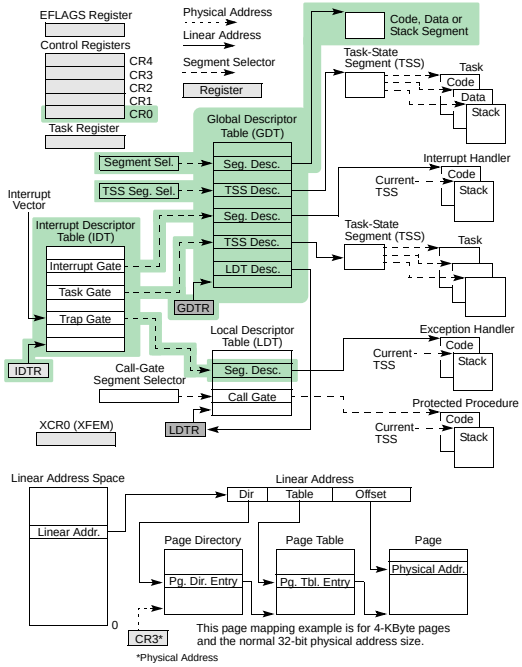
David Alejandro González Márquez

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

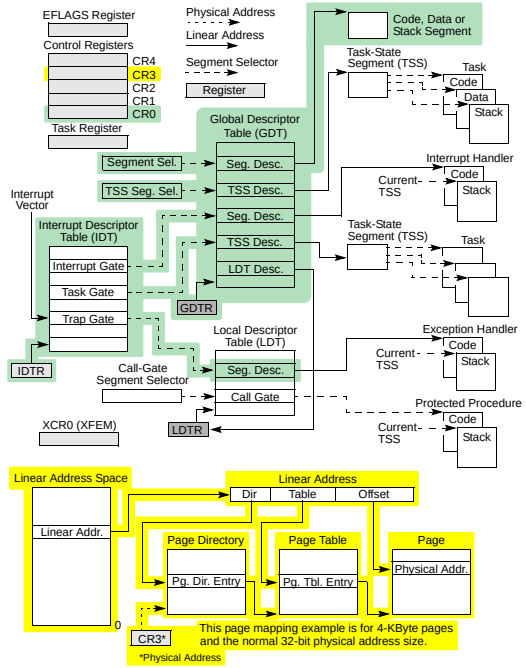
Usted ...


EFLAGS Register
Control Registers
CR4
CR3
CR2
CR1
CR0
Task Register
Physical Address
Linear Address
Segment Selector
Register
Global Descriptor Table (GDT)
Segment Sel.
TSS Seg. Sel.
Seg. Desc.
TSS Desc.
Seg. Desc.
TSS Desc.
LDT Desc.
GDTR
Interrupt Vector
Interrupt Descriptor Table (IDT)
Interrupt Gate
Task Gate
Trap Gate
IDTR
Local Descriptor Table (LDT)
Seg. Desc.
Call Gate
Call-Gate Segment Selector
LDTR
XCR0 (XFEM)
Code, Data or Stack Segment
Task-State Segment (TSS)
Task
Code
Data
Stack
Interrupt Handler
Code
Current-TSS
Stack
Task-State Segment (TSS)
Task
Exception Handler
Code
Current-TSS
Stack
Protected Procedure
Code
Current-TSS
Stack
Linear Address Space
Linear Addr.
0
CR3*
Linear Address
Dir
Table
Offset
Page Directory
Pg. Dir. Entry
Page Table
Pg. Tbl. Entry
Page
Physical Addr.
This page mapping example is for 4-KByte pages and the normal 32-bit physical address size.
*Physical Address

Usted estabá aquí

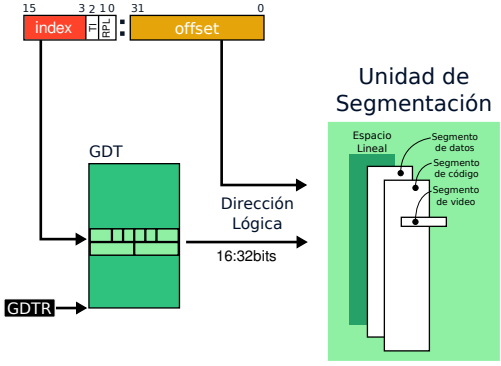


Usted estará aquí

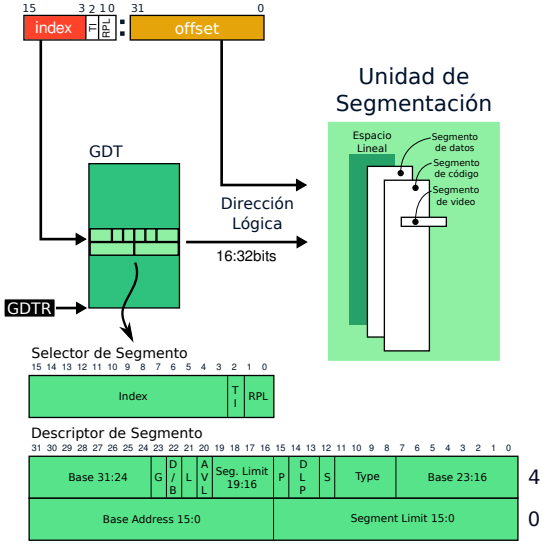


Unidades de administación de memoria

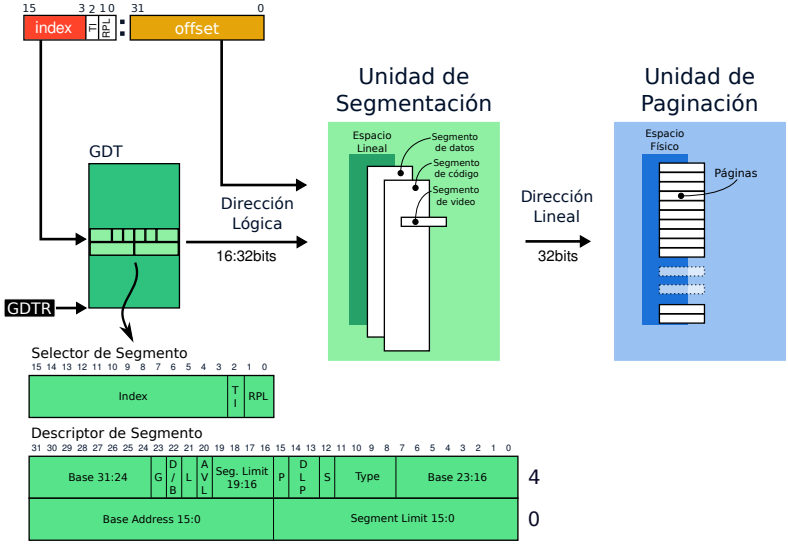
Unidades de administación de memoria



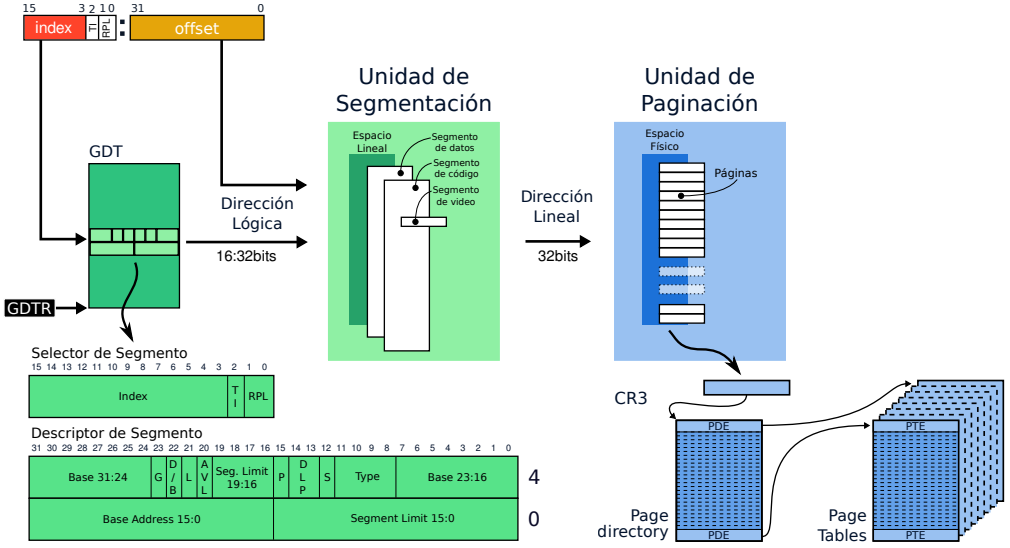
Unidades de administación de memoria



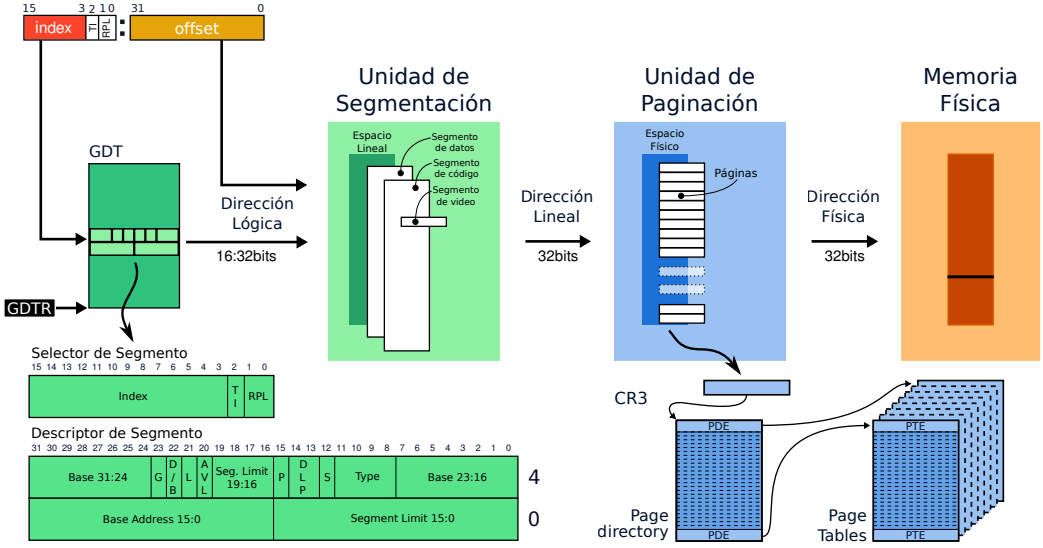
Unidades de administación de memoria



Unidades de administación de memoria

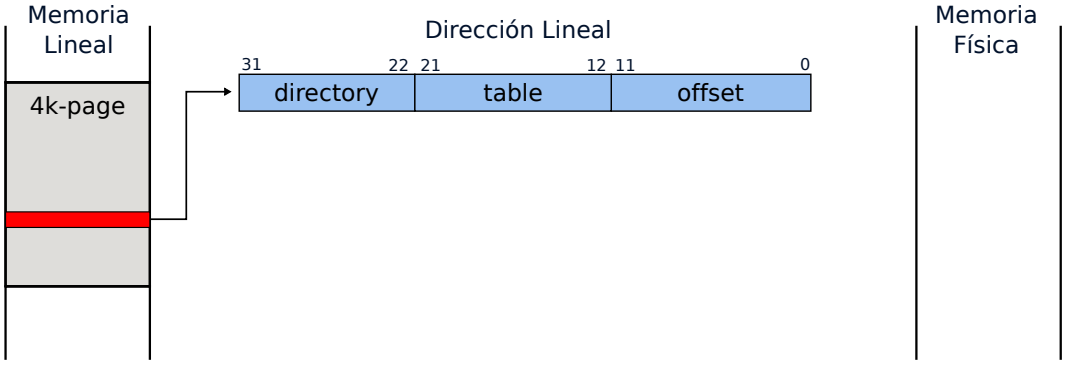


Unidades de administación de memoria

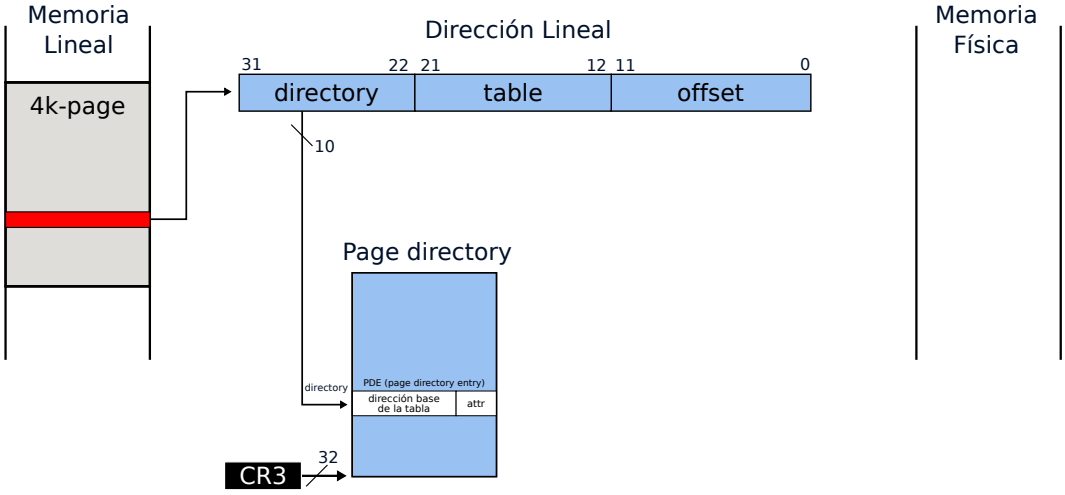


Mecanismo de Paginación

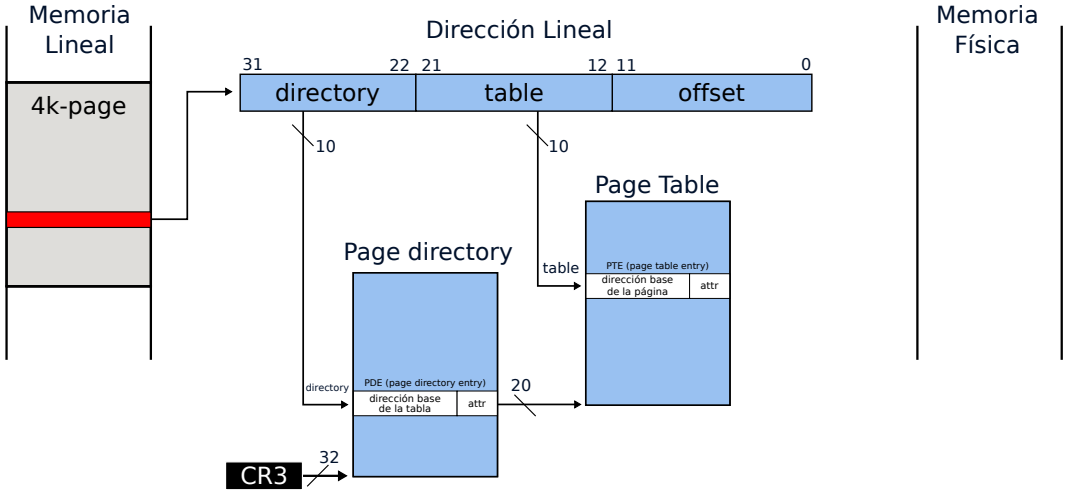
Mecanismo de Paginación



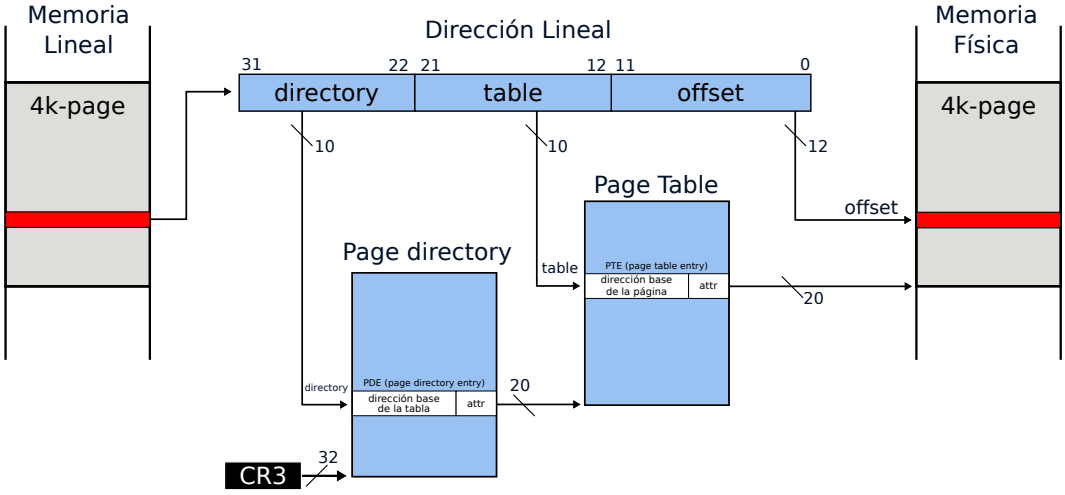
Mecanismo de Paginación



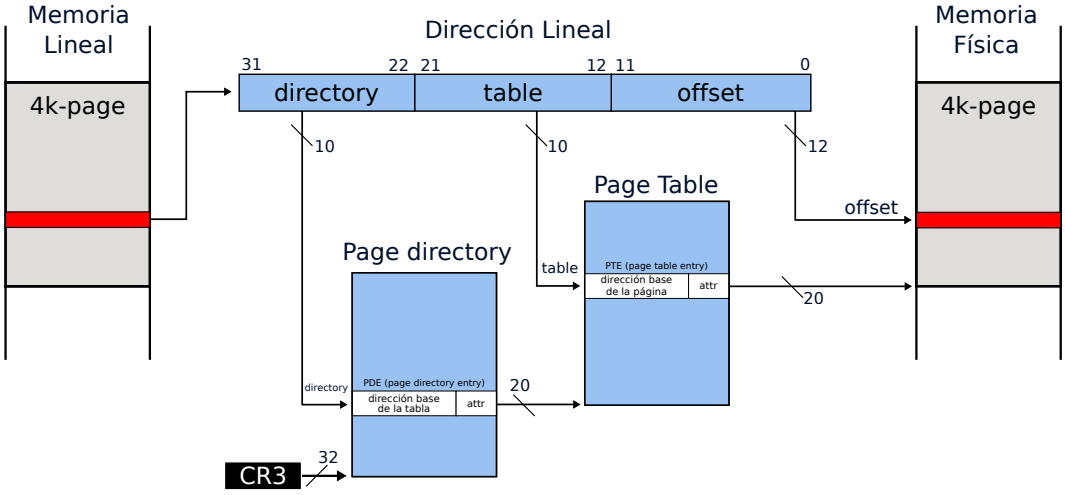
Mecanismo de Paginación



Mecanismo de Paginación



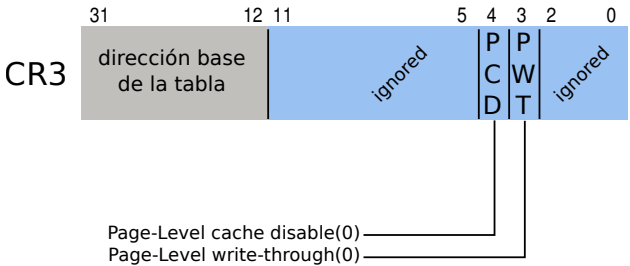
Mecanismo de Paginación



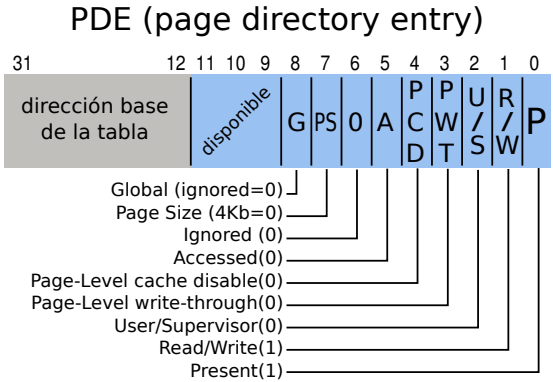
Para el contexto del TP vamos a limitarnos a Paginación con páginas de 4KB sin PAE

CR3 - Control Register 3

- Puntero a la base de la page `directory`.
- Modificar este registro implica resetear la TLB.

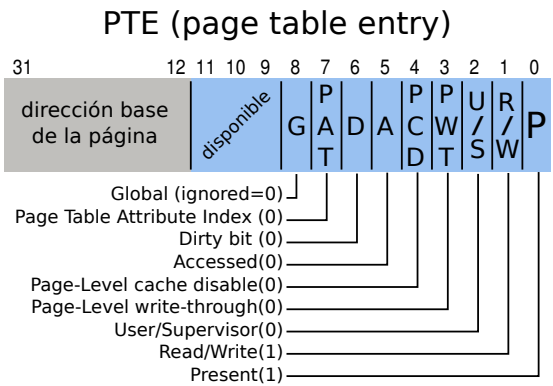


PDE - Page Directory Entry



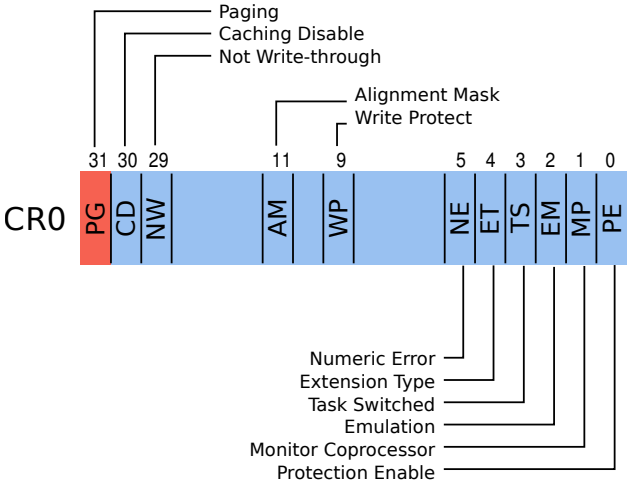
- Entrada de la page `directory`
- Atributos importantes:
  - `Presente`
  - `Read/Write`
  - `User/Supervisor`

PDE - Page Table Entry



- Entrada de la page `table`
- Atributos importantes:
  - `Presente`
  - `Read/Write`
  - `User/Supervisor`

CR0 - Control Register 0

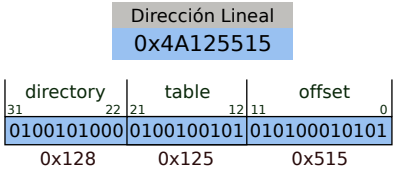


- Permite activar Paginación

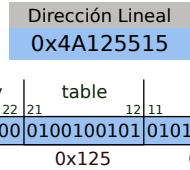
Ejemplo Numérico

Dirección Lineal
0x4A125515

Ejemplo Numérico



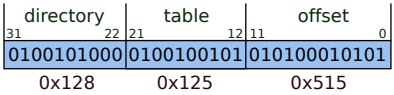
Ejemplo Numérico

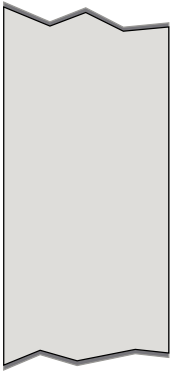


Ejemplo Numérico

Dirección Lineal

0x4A125515



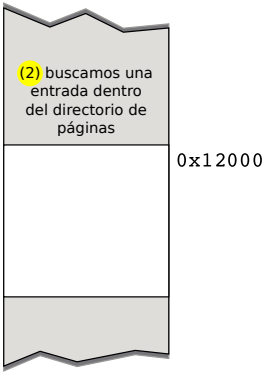


Ejemplo Numérico

Dirección Lineal

0x4A125515

directory	table	offset
31 22	21 12	11 0
0100101000	0100100101	010100010101
0x128	0x125	0x515

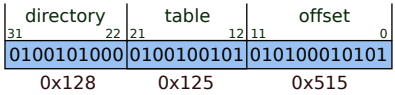


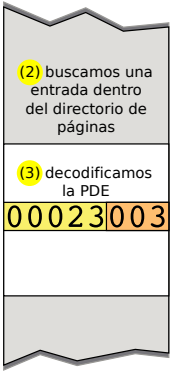
Page Directory

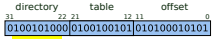
Ejemplo Numérico

Dirección Lineal

0x4A125515







0x12000

0x12000+0x128*4

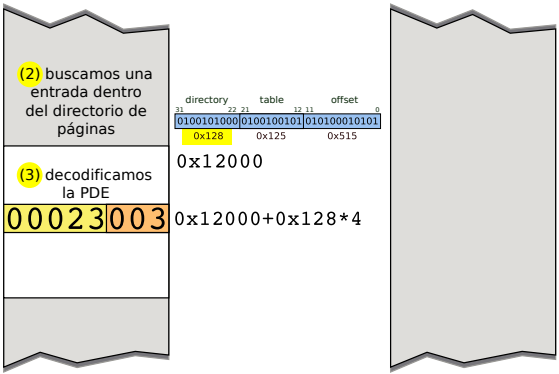
Page Directory

Ejemplo Numérico

Dirección Lineal

0x4A125515

directory	table	offset
31 22	21 12	11 0
0100101000	0100100101	010100010101
0x128	0x125	0x515



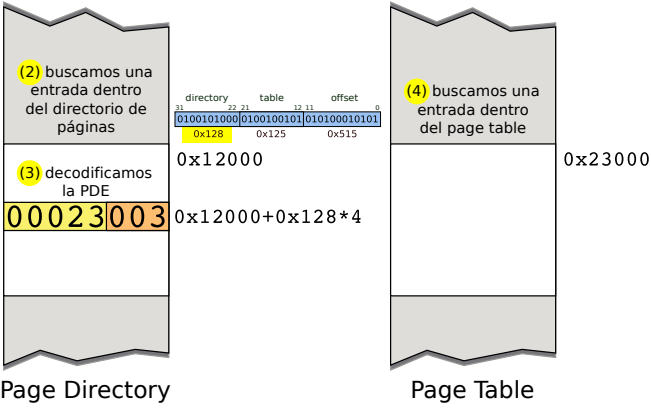
Page Directory

Ejemplo Numérico

Dirección Lineal

0x4A125515

directory	table	offset
31 22	21 12	11 0
0100101000	0100100101	010100010101
0x128	0x125	0x515

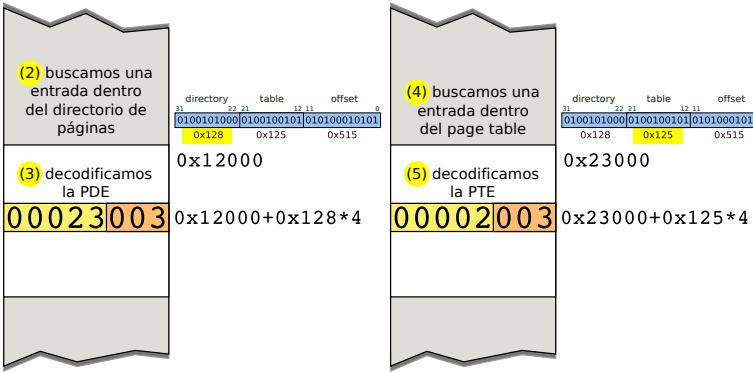


Ejemplo Numérico

Dirección Lineal

0x4A125515

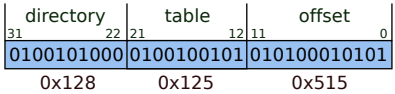
directory	table	offset
31 22	21 12	11 0
0100101000	0100100101	010100010101
0x128	0x125	0x515

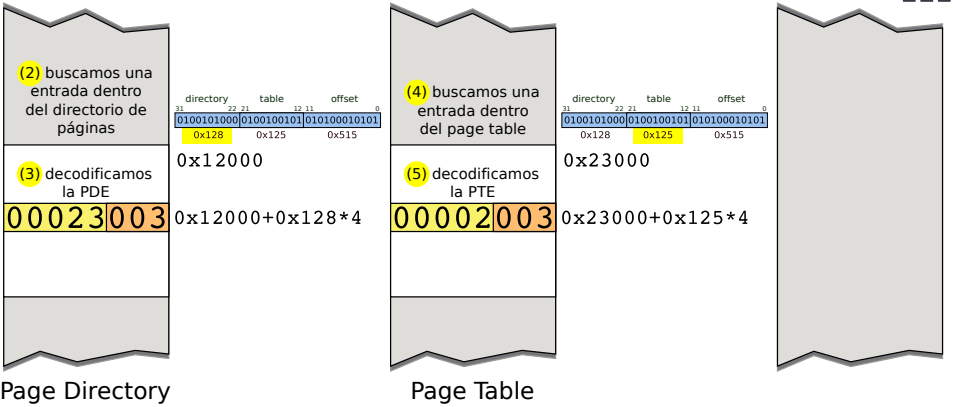


Ejemplo Numérico

Dirección Lineal

0x4A125515





Ejemplo Numérico

Dirección Lineal

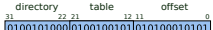
0x4A125515

directory	table	offset
31 22	21 12	11 0
0100101000	0100100101	010100010101
0x128	0x125	0x515

(1) buscar el CR3

CR3 00012000

(2) buscamos una entrada dentro del directorio de páginas



0x12000

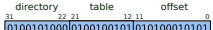
(3) decodificamos la PDE

00023003

0x12000+0x128*4

Page Directory

(4) buscamos una entrada dentro del page table



0x23000

(5) decodificamos la PTE

00002003

0x23000+0x125*4

Page Table

0x02000

4k-page

Ejemplo Numérico

Dirección Lineal
0x4A125515

directory	table	offset
31 22	21 12	11 0
0100101000	0100100101	010100010101
0x128	0x125	0x515

(1) buscar el CR3

CR3 00012000

(2) buscamos una entrada dentro del directorio de páginas

directory table offset
0100101000 0100100101 010100010101
0x128 0x125 0x515

0x12000

(3) decodificamos la PDE

00023003

0x12000+0x128*4

Page Directory

(4) buscamos una entrada dentro del page table

directory table offset
0100101000 0100100101 010100010101
0x128 0x125 0x515

0x23000

(5) decodificamos la PTE

00002003

0x23000+0x125*4

Page Table

directory table offset
0100101000 0100100101 010100010101
0x128 0x125 0x515

0x02000

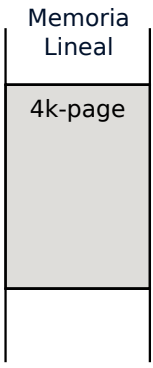
C7

0x02000+0x515

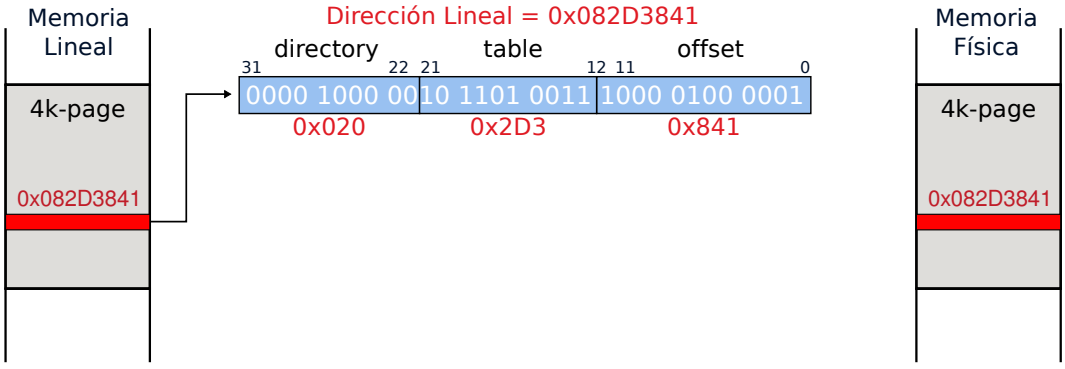
(6) ¡al fin! ahora buscamos la entrada de memoria que queremos

4k-page

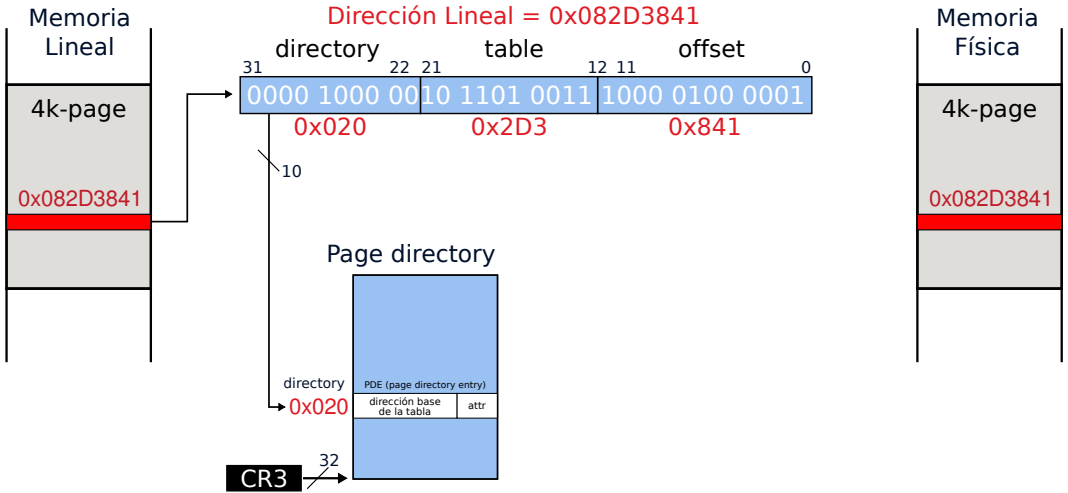
Ejemplo Identity Mapping



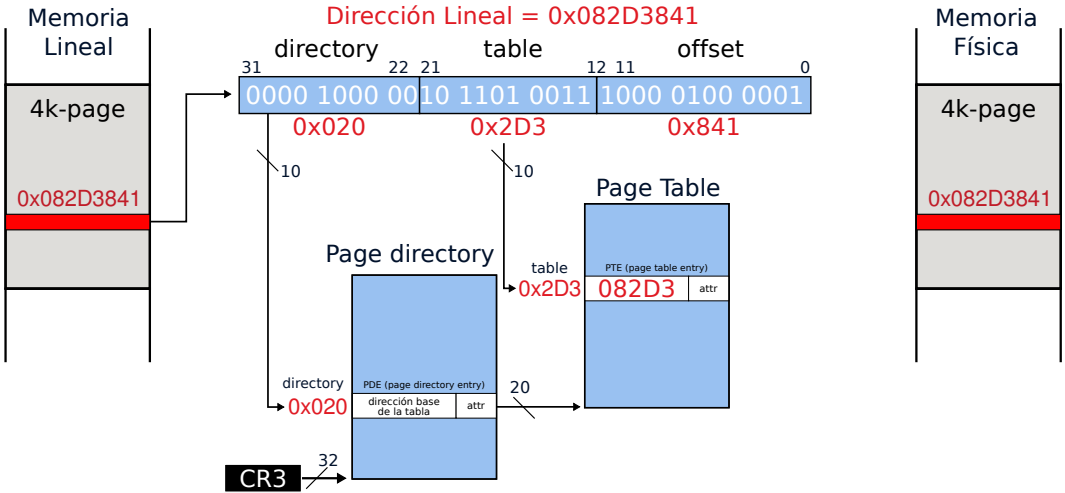
Ejemplo Identity Mapping



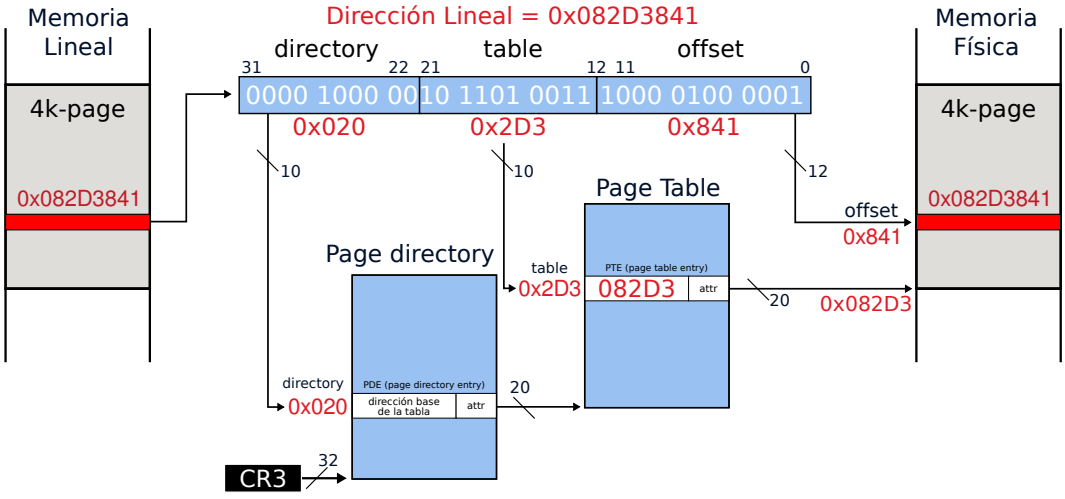
Ejemplo Identity Mapping



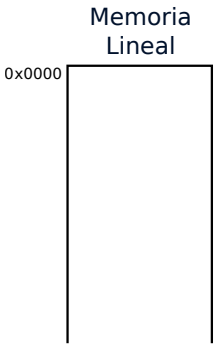
Ejemplo Identity Mapping

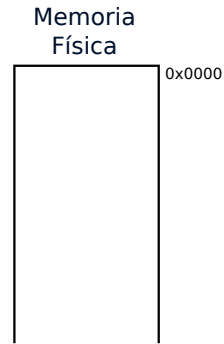


Ejemplo Identity Mapping

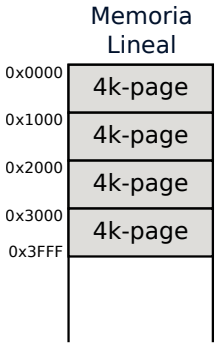


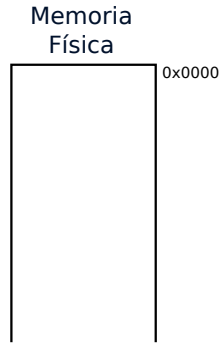
Identity Mapping para los primeros 16KB





Identity Mapping para los primeros 16KB

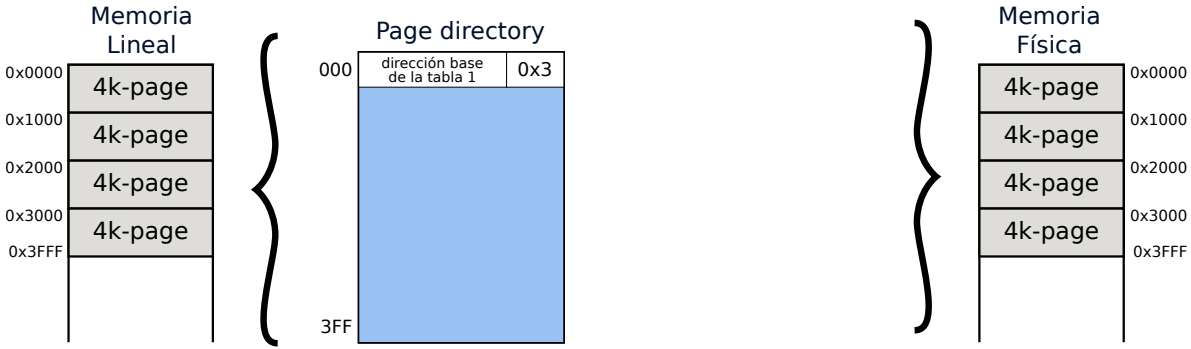




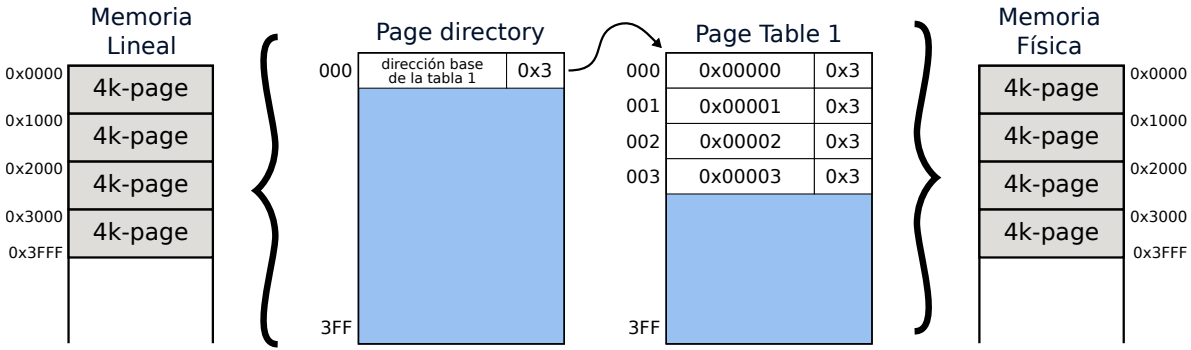
Identity Mapping para los primeros 16KB

Identity Mapping para los primeros 16KB

Identity Mapping para los primeros 16KB



Identity Mapping para los primeros 16KB



Resolver direcciones

Segmentación

DS	Base =	0x00000000
	Límite =	0xFFFFF
	G =	1

ES	Base =	0x000B8000
	Límite =	0x01F3F
	G =	0

Resolver direcciones

Segmentación

DS	Base =	0x00000000
	Límite =	0xFFFFF
	G =	1

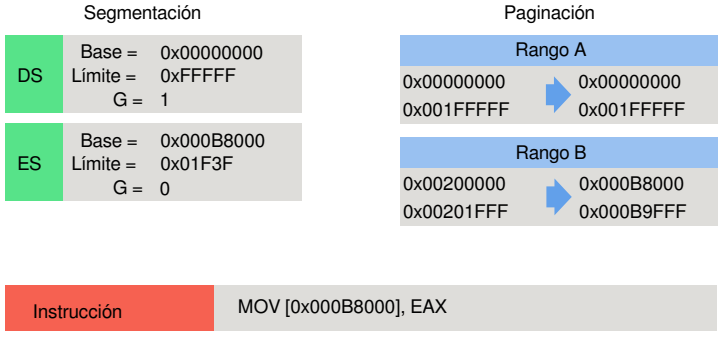
ES	Base =	0x000B8000
	Límite =	0x01F3F
	G =	0

Paginación

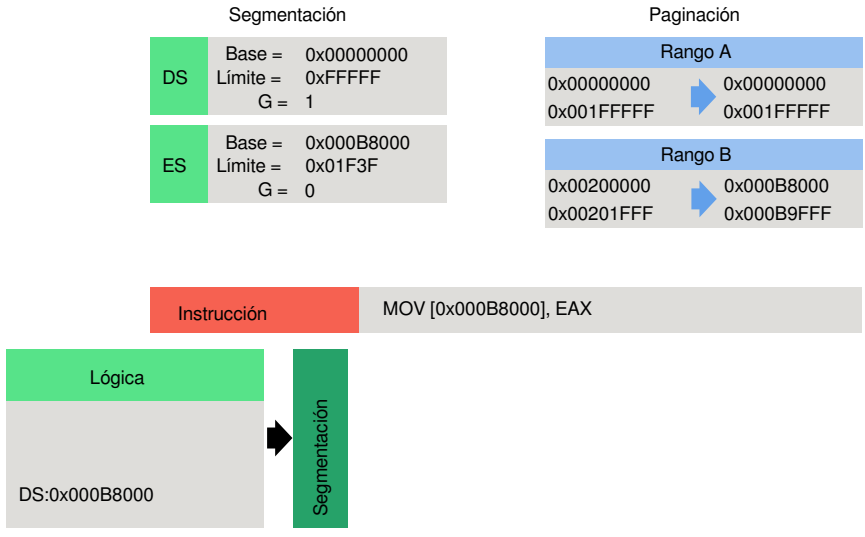
Rango A	
0x00000000 0x001FFFFF	0x00000000 0x001FFFFF

Rango B	
0x00200000 0x00201FFF	0x000B8000 0x000B9FFF

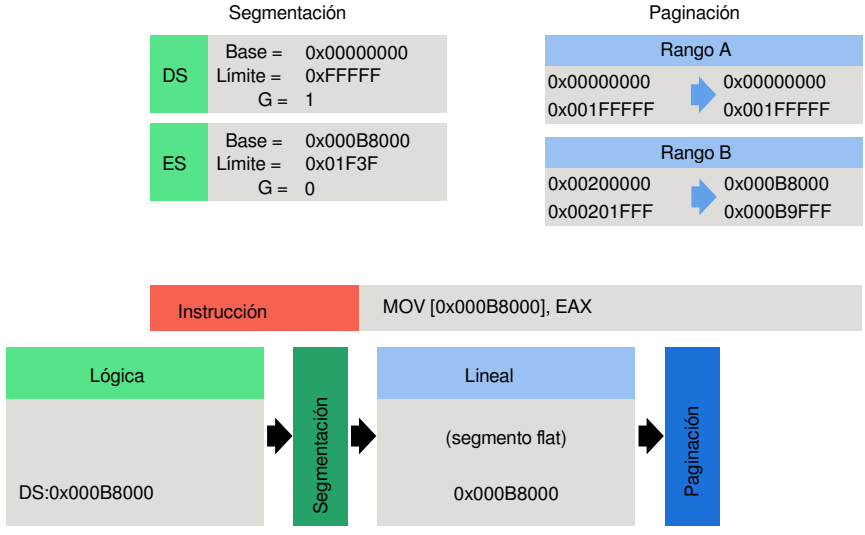
Identity Mapping en segmentación y paginación



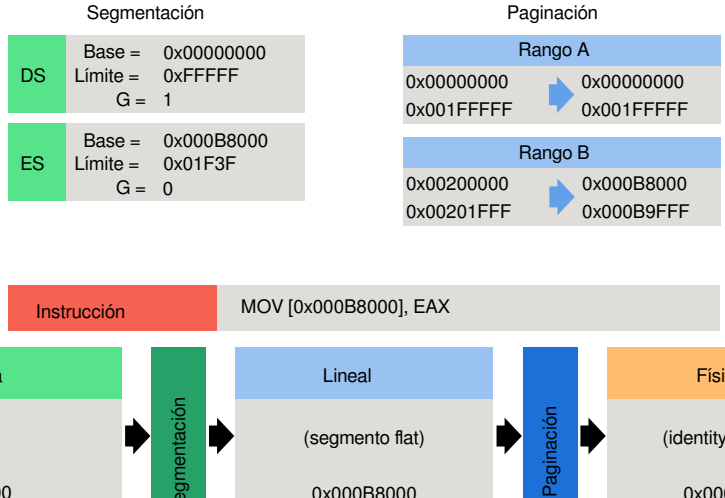
Identity Mapping en segmentación y paginación



Identity Mapping en segmentación y paginación



Identity Mapping en segmentación y paginación



Identity Mapping en segmentación, paginación distinta de Identity Mapping

Segmentación

DS	Base =	0x00000000
	Límite =	0xFFFFF
	G =	1

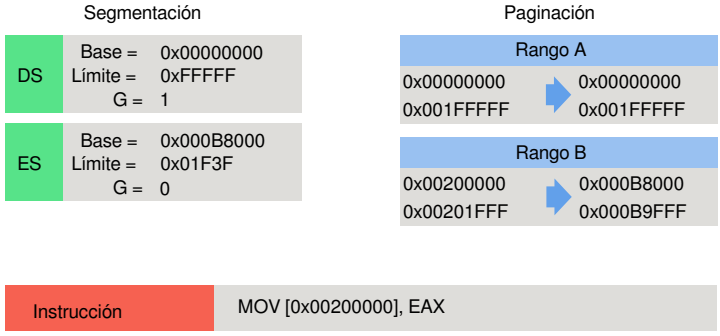
ES	Base =	0x000B8000
	Límite =	0x01F3F
	G =	0

Paginación

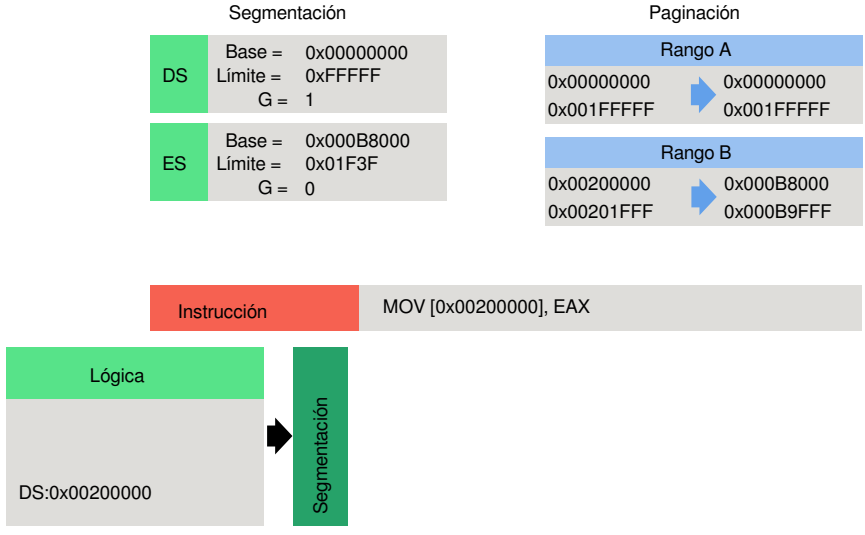
Rango A	
0x00000000 0x001FFFFF	0x00000000 0x001FFFFF

Rango B	
0x00200000 0x00201FFF	0x000B8000 0x000B9FFF

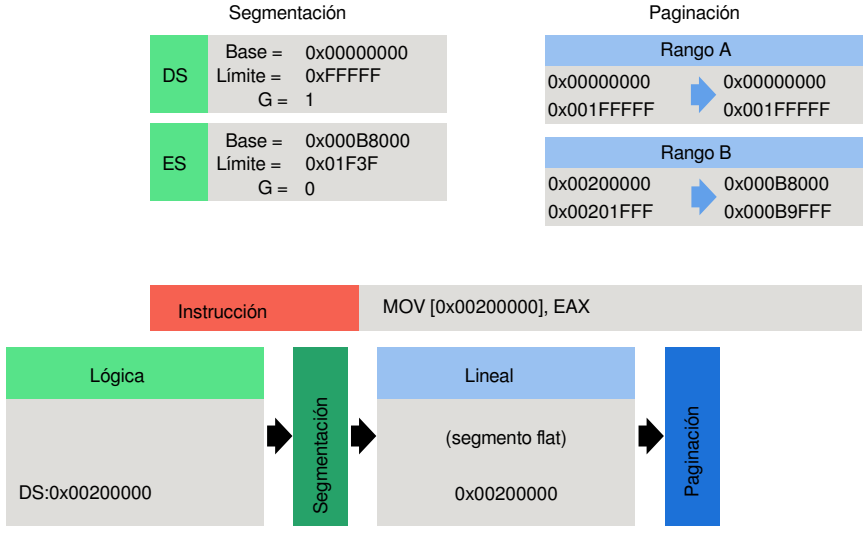
Identity Mapping en segmentación, paginación distinta de Identity Mapping



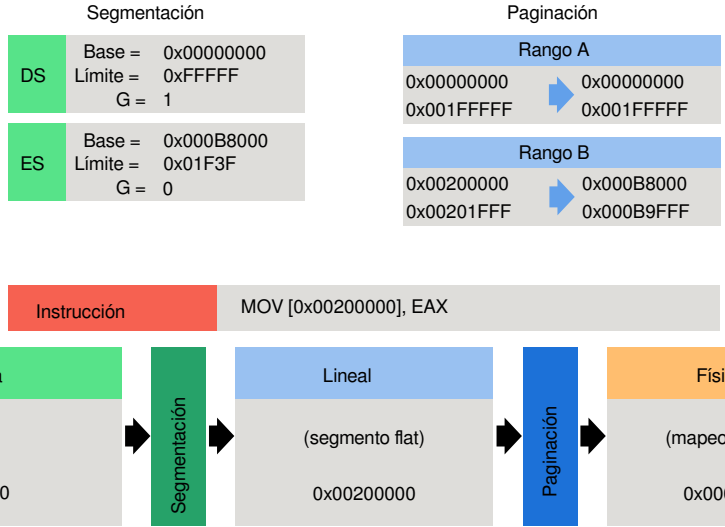
Identity Mapping en segmentación, paginación distinta de Identity Mapping



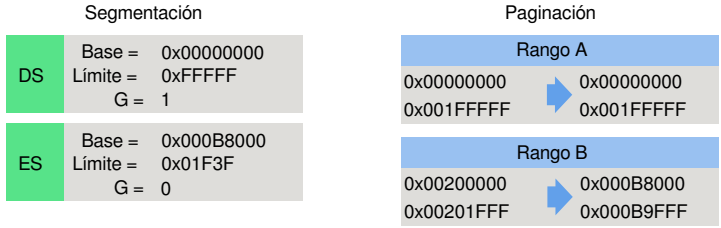
Identity Mapping en segmentación, paginación distinta de Identity Mapping



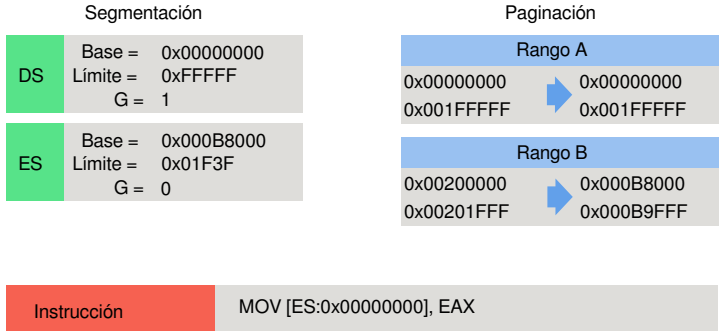
Identity Mapping en segmentación, paginación distinta de Identity Mapping



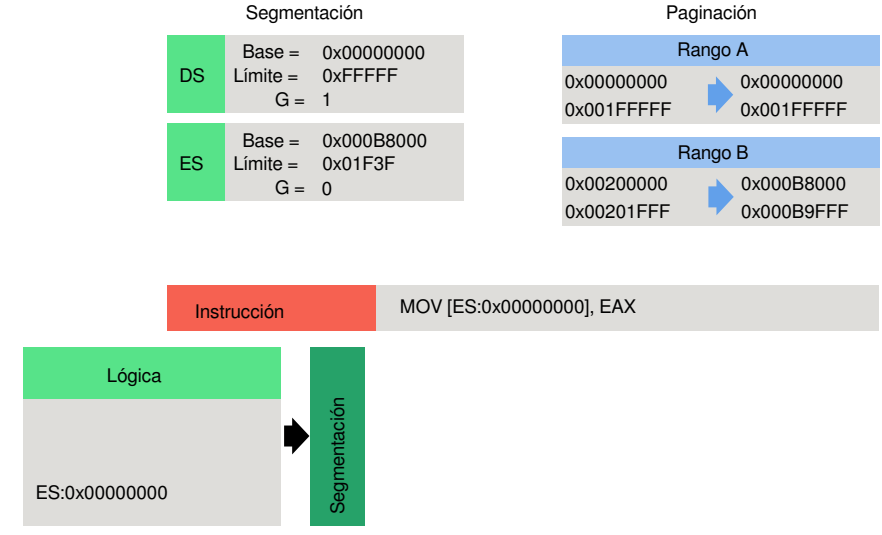
Identity Mapping en paginación, segmentación con base distinta de cero



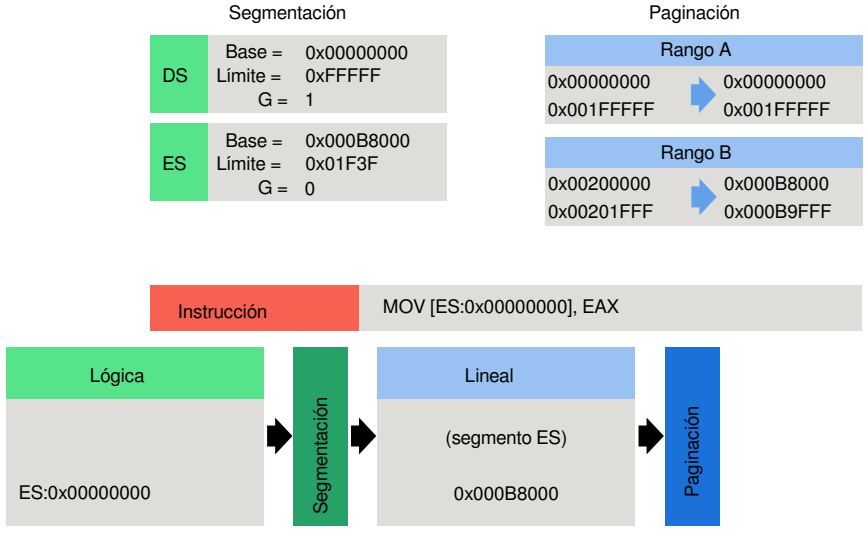
Identity Mapping en paginación, segmentación con base distinta de cero



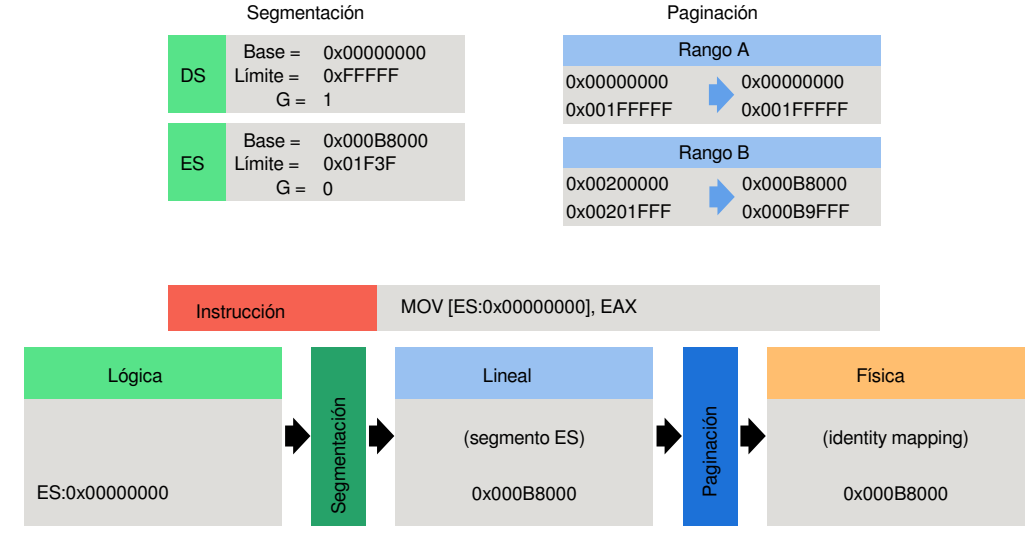
Identity Mapping en paginación, segmentación con base distinta de cero



Identity Mapping en paginación, segmentación con base distinta de cero



Identity Mapping en paginación, segmentación con base distinta de cero



Hasta ahora tenemos segmentación activa,

- Deshabilitamos las interrupciones `CLI`
- Habilitamos la `A20`
- Creamos y completamos la `GDT`
- Cargamos el registro `GDTR` con la dirección base y el límite de la `GDT`
- Seteamos el bit PE del registro `CR0`

Hasta ahora tenemos segmentación activa,

- Deshabilitamos las interrupciones `CLI`
- Habilitamos la `A20`
- Creamos y completamos la `GDT`
- Cargamos el registro `GDTR` con la dirección base y el límite de la `GDT`
- Seteamos el bit `PE` del registro `CR0`
- Realizamos un `JUMP FAR` a la siguiente instrucción
- Actualizamos la información de los registros de segmento DS, ES, GS, FS y SS
- Pintamos la pantalla

Tenemos además interrupciones,

- Creamos y completamos una `IDT` básica
- Asociamos cada excepción a su rutina de atención
- Asociamos las rutinas de interrupciones externas

Tenemos además interrupciones,

- Creamos y completamos una `IDT` básica
- Asociamos cada excepción a su rutina de atención
- Asociamos las rutinas de interrupciones externas
- Creamos entradas para los servicios del sistema
- Programamos la rutina de atención de reloj
- Programamos la rutina de atención de teclado

Y ahora activamos paginación,

- Armar un directorio de páginas en `0x27000` y una tablas de páginas en `0x28000`. Mapeando los primeros 4MB de memoria con Identity Mapping.

Y ahora activamos paginación,

- Armar un directorio de páginas en `0x27000` y una tablas de páginas en `0x28000`. Mapeando los primeros 4MB de memoria con Identity Mapping.

- Cargar en el registro CR3 la dirección base del directorio de páginas.

Y ahora activamos paginación,

- Armar un directorio de páginas en `0x27000` y una tablas de páginas en `0x28000`. Mapeando los primeros 4MB de memoria con Identity Mapping.

- Cargar en el registro CR3 la dirección base del directorio de páginas.

- Limpiar los bits PCD y PWT del registro CR3.

Y ahora activamos paginación,

- Armar un directorio de páginas en `0x27000` y una tablas de páginas en `0x28000`. Mapeando los primeros 4MB de memoria con Identity Mapping.
- Cargar en el registro CR3 la dirección base del directorio de páginas.
- Limpiar los bits PCD y PWT del registro CR3.
- Setear el bit PG de CR0 para activar paginación.

Y ahora activamos paginación,

- Armar un directorio de páginas en `0x27000` y una tablas de páginas en `0x28000`. Mapeando los primeros 4MB de memoria con Identity Mapping.
- Cargar en el registro CR3 la dirección base del directorio de páginas.
- Limpiar los bits PCD y PWT del registro CR3.
- Setear el bit PG de CR0 para activar paginación.

A partir de ahora TODOS los accesos a memoria serán resueltos usando paginación

Seudocódigo para armar Indentity Mapping en el Kernel

```
mmu_initKernelDir()

    PD = KERNEL_PAGE_DIR
    PT = KERNEL_PAGE_TABLE_0

    mmu_zerosPage(PD)
    mmu_zerosPage(PT)

    PD[0] = PT | PAG_S | PAG_RW | PAG_P

    for (i = 0; i < 1024; i++)
        PT[i] = (i << 12) | PAG_S | PAG_RW | PAG_P

    return PD
```

Código del Kernel para activar Paginación

```
; Inicializar el manejador de memoria
call mmu_init

; Inicializar el directorio de paginas
call mmu_initKernelDir

; Cargar directorio de paginas
mov cr3, eax

; Habilitar paginacion
mov eax, cr0
or  eax, 0x80000000
mov cr0, eax
```

Bibliografía: Fuentes y material adicional

- Convenciones de llamados a función en x86:
  `https://en.wikipedia.org/wiki/X86_calling_conventions`
- Notas sobre System V ABI:
  `https://wiki.osdev.org/System_V_ABI`
- Documentación de NASM:
  `https://nasm.us/doc/`
- Artículo sobre el flag `-pie`:
  `https://eklitzke.org/position-independent-executables`
- Documentación de System V ABI:
  `https://uclibc.org/docs/psABI-x86_64.pdf`
- Manuales de Intel:
  `https://software.intel.com/en-us/articles/intel-sdm`

¡Gracias!

Recuerden leer los comentarios al final de este video por aclaraciones o fe de erratas.