Ejercicios pre parcial tipo 1

Organización del Computador II

Facundo Ruiz

Departamento de Computación - FCEyN UBA

Segundo cuatrimestre de 2019

Hoy

- La clase de hoy se centrará en hacer ejercicios de parcial relacionados con **segmentación** y **paginación**
- Nos centraremos específicamente en cómo manejar las **estructuras** que permiten resolver las direcciones
- Ahora **¿cómo se resuelven las direcciones?**

  $$\text{Lógica} \xrightarrow[\text{Segmentación}]{} \text{Lineal} \xrightarrow[\text{Paginación}]{} \text{Física}$$

- **¿Qué estructuras son necesarias para la segmentación?**

  GDT / LDT

- **¿Y para la paginación?**

  Page Directory (PD) y Page Table (PT)

Segmentación: Repaso de selectores

Repasemos ahora los mecanismos asociados a segmentación:

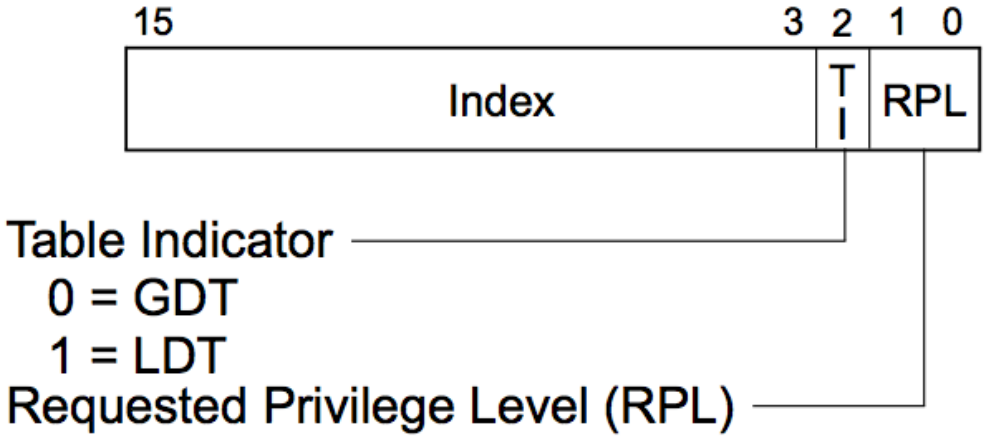
- **¿Cómo accedemos a un segmento?**

  A través de una dirección lógica

- **¿Qué forma tiene esta dirección?**

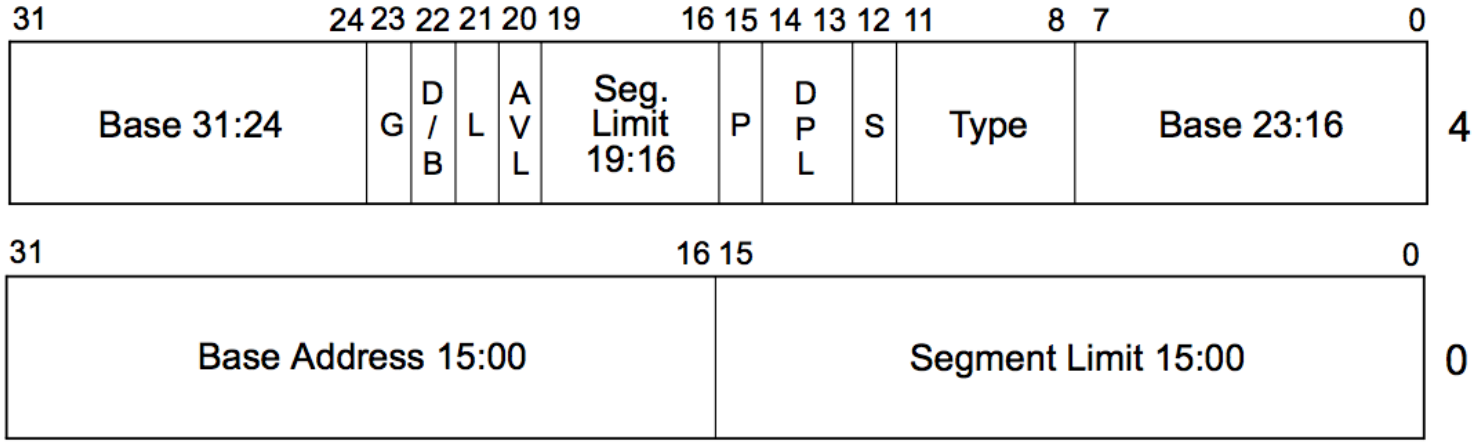
  `(selector-segmento : offset)`

- **¿Cómo sabemos a qué segmento estamos referenciando?**
  Con el selector vemos la tabla en que está descripto (`TI`) y la entrada que lo describe (`index`)



Segmentación: Descriptor GDT

En base a un selector llegamos a un descriptor de segmento:



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

Segmentación: Repaso de descriptores (I)

Veamos cómo se describe el segmento en base a sus campos:

- **¿Cómo sabemos si el segmento está presente?**

  Vemos si el bit P de su descriptor está en 1

- **¿Y para saber dónde comienza en memoria?**

  Vemos su campo base completo (uniendo sus partes)

- **¿Qué es el campo límite (uniendo sus partes)?**

  Indica el máximo desplazamiento dentro de un segmento junto con el bit G

- **¿Cuánto vale el máximo desplazamiento si G está en 1?**

  Podemos determinarlo de varias maneras:

  - `((límite + 1) * 4kb) - 1`
  - `((límite + 1) << 12) - 1`
  - `(límite << 12) + 0xFFF`

- En base a estos datos determinamos el **espacio** que ocupa el segmento en memoria

Segmentación: Repaso de descriptores (II)

Veamos otros campos importantes:

- **¿Qué campos determinan el tipo del segmento?**

  Los campos `type` y `S`

- **¿Y su nivel de acceso?**

  El campo `DPL` (veremos más en detalle cómo es esta verificación en la clase de Protección)

- **¿Qué hay del campo D/B?**

  Determina el tamaño por defecto de las operaciones (suele estar en 1 para 32 bits)

- **¿Y los campos L y AVL?**

  Si bien tienen su uso, en nuestro caso van a estar en 0

Segmentación: Tipos de Entradas

Considerando un segmento que no es de sistema ($S = 1$):

Type Field					Descriptor Type	Description
Decimal	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

De segmentación a paginación

Con la descripción del segmento podemos determinar la dirección lineal resultante de la lógica de la que partimos:

- **¿Cómo hacemos esta traducción?**

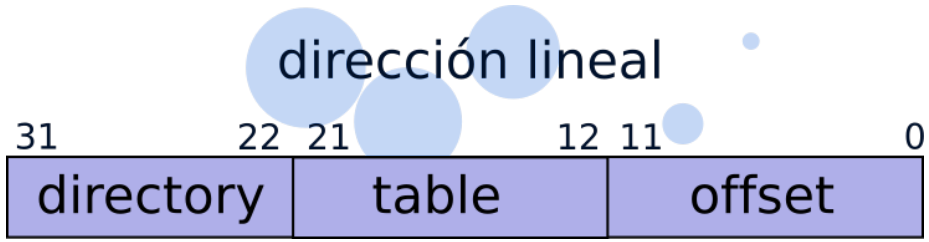
  Tomamos la dirección base del descriptor (juntando sus partes) y le sumamos el offset

- **¿Qué uso tiene la dirección resultante?**

  Permite determinar la página física a la que queremos acceder

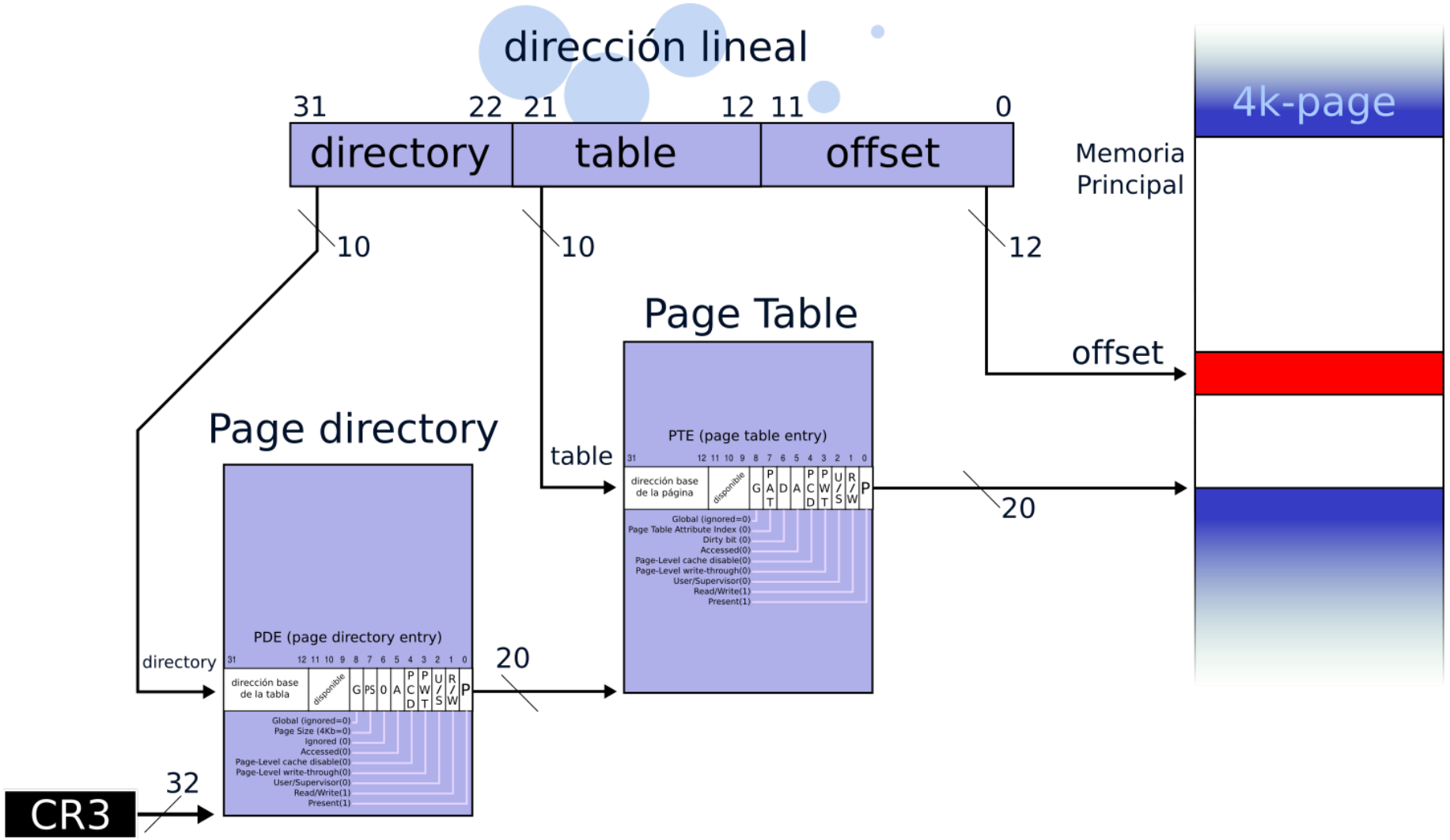
- **¿Y cómo la determinamos?**

  A través de los índices en que se divide la dirección:



© David

De una dirección virtual a una física



Paginación: Repaso de descriptores (I)

Recordemos algunas cuestiones referentes a paginación:

- **¿Dónde está el directorio de páginas en memoria?**

  En una dirección dada por el registro `CR3`

- **¿Qué describe cada una de sus entradas (PDE)?**

  Los accesos a una tabla de páginas

- **¿Y cada entrada de una tabla de páginas (PTE)?**

  Los accesos a una página física (distintas PTE pueden describir la misma página física)

- **¿Cómo se describen estas entradas?**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Address of 4KB page frame																				Ignored			G	PAT	D	A	PCD	PWT	U/S	R/W	1	**PTE: 4KB page**
Address of page table																				Ignored				0	Ign	A	PCD	PWT	U/S	R/W	1	**PDE: page table**

Paginación: Repaso de descriptores (II)

Repasemos los campos importantes de estos descriptores:

- **¿Cómo sabemos si la pagína está presente en memoria?**

  Vemos si los bits P de su PDE y PTE están en 1
- **¿Y para saber si su acceso es de nivel usuario?**

  Vemos si los bits U/S de su PDE y PTE están en 1
- **¿Qué sucede si alguno está en 0?**

  La página es de nivel supervisor
- **¿Y para ver si es escribible?**

  Vemos si los bits R/W de su PDE y PTE están en 1

Ejercicio 1

Consigna

Dada la siguiente tabla de traducciones, dar un conjunto de descriptores de segmento, directorio de páginas y tablas de páginas que cumplan con todas las traducciones **en simultáneo**.
Completar todos sus campos indicando si hay **identity mapping** en cada acceso

Lógica	Lineal	Física	Características
`0x0023:0x000000FF`	`0x10192FFF`	`0x00AAEFFF`	Datos Nivel 3 Acción: Lectura de 4 bytes
`0x0320:0x00000001`	`0xFF07A011`	`0x0391F011`	Código Nivel 0 Acción: Lectura de 2 bytes
`0x0320:0x00001233`	`0xFF07B243`	`0x003A2243`	Código Nivel 0 Acción: Ejecución de 1 bytes
`0x0411:0x00004FFF`	`0x38442341`	`0x00333341`	Datos Nivel 2 Acción: Escritura de 4 bytes

Ejercicio 2

Consigna

Dada la siguiente gdt, directorio y tablas de páginas (simplificados), completar la dirección física de cada uno de los accesos indicados. Identificar en cada uno si hay **identity mapping**.

GDT	Base	Atributos
0x002	0x00000200	...
0x034	0x00011000	...
0x104	0x04400001	...

PD	Base	Atributos
0x000	PT_1 >> 12	...
0x011	PT_2 >> 12	...

PT_1	Base	Atributos
0x001	0x73682	...
0x141	0x12873	...

PT_2	Base	Atributos
0x000	0x77777	...
0x001	0x0DEAD	...
0x077	0x32343	...
0x078	0x0CAFE	...

Dirección lógica	Dirección física
0x0010:0x00141CFE	
0x0820:0x00000123	
0x01A0:0x04466AB1	
0x0820:0x0007800E	

Eso es todo por hoy

¿Preguntas?