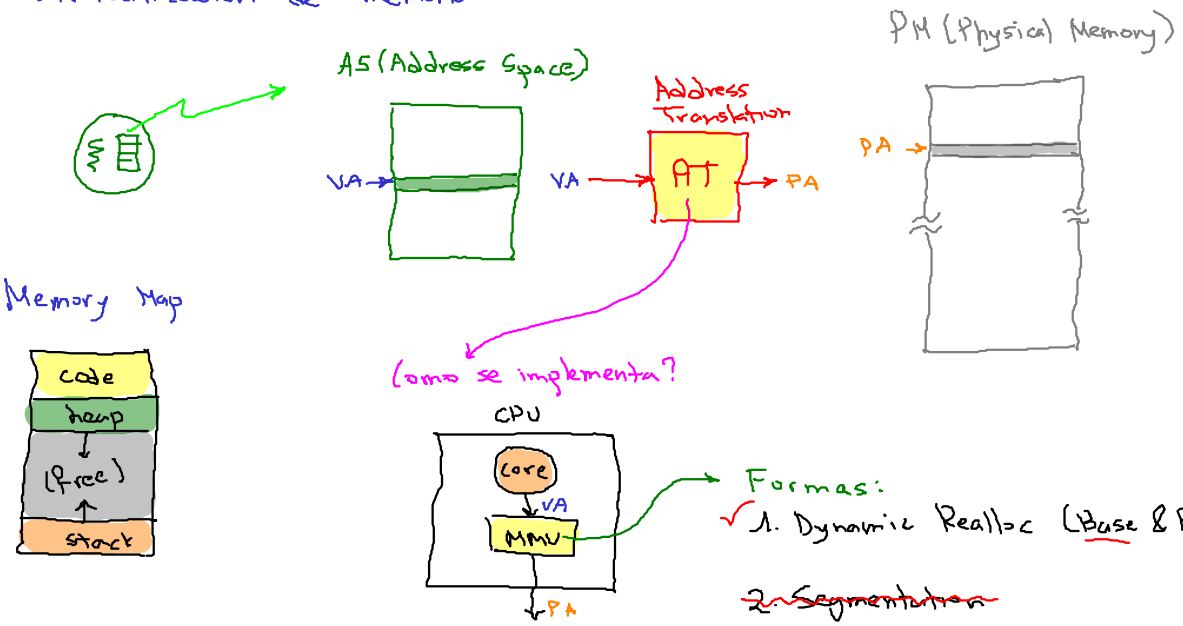
# 06/05/2025 – Sistemas Operativos – Ude@

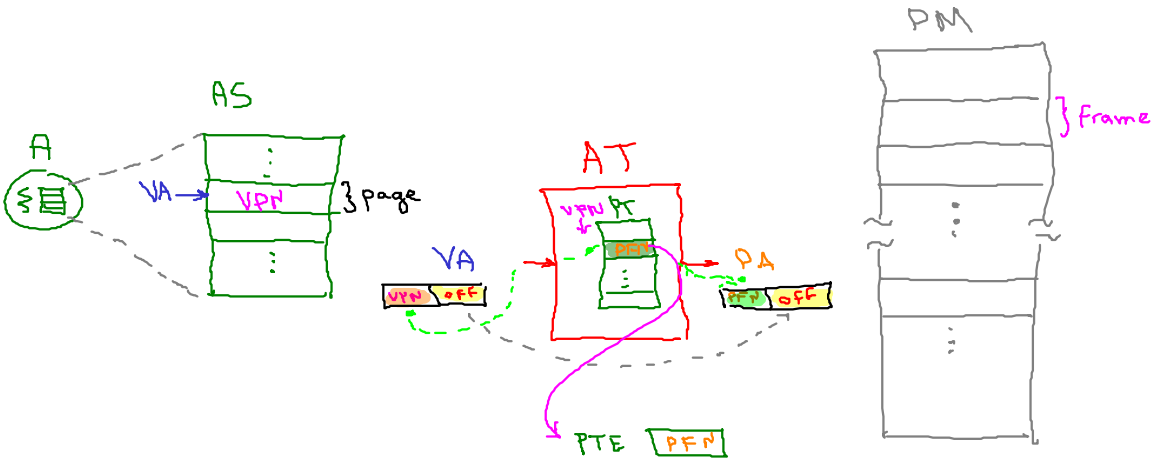
## 1. Repaso – Virtualización de memoria




Formas:

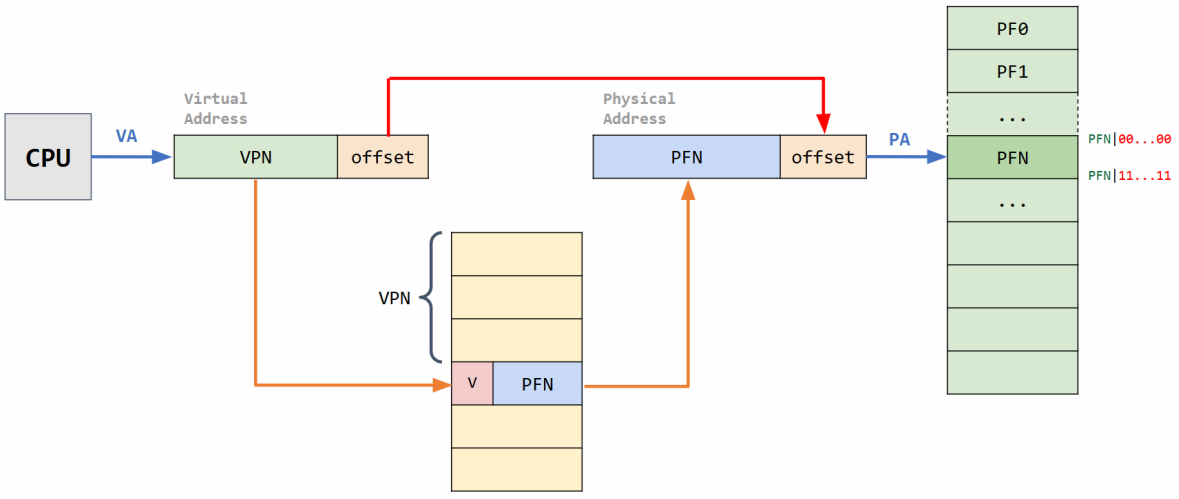
- ✓ 1. Dynamic Realloc (Base & Bound)
- ~~2. Segmentation~~
- ✓ 3. Paginación

## 2. Paginación.

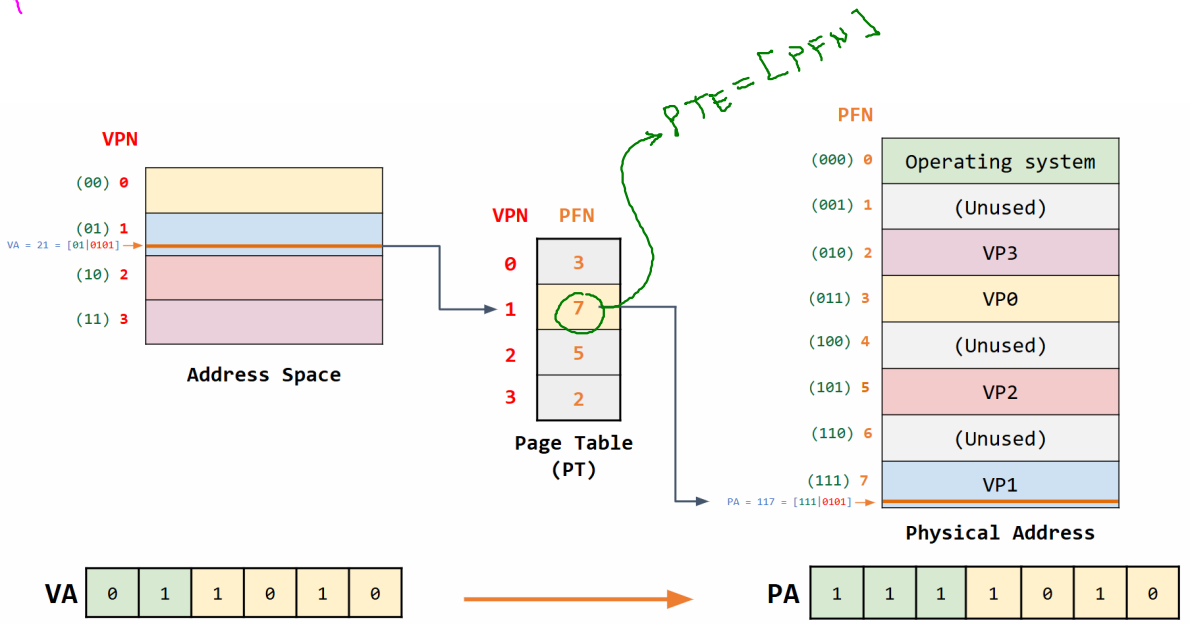



- PFN: Page Frame Number
- VPN: Virtual page Number

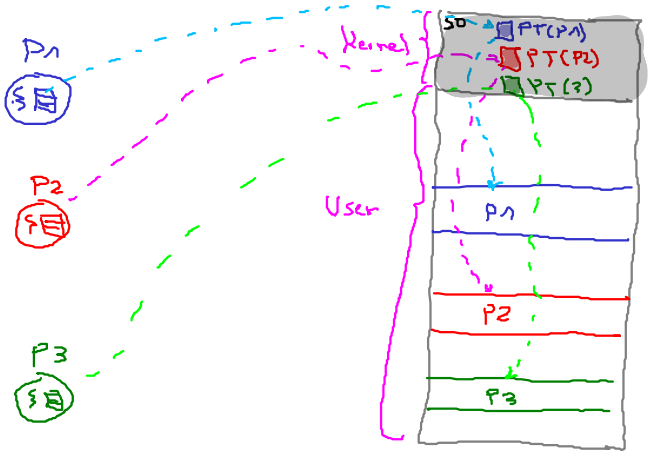
Mecanismo de paginación:



Ejemplo anterior:




3. Overhead por PT. → Problema: Overhead de gasto de memoria por las PTs.




**Ejemplo**:

Cual es el impacto en memoria cuando se usa paginación?

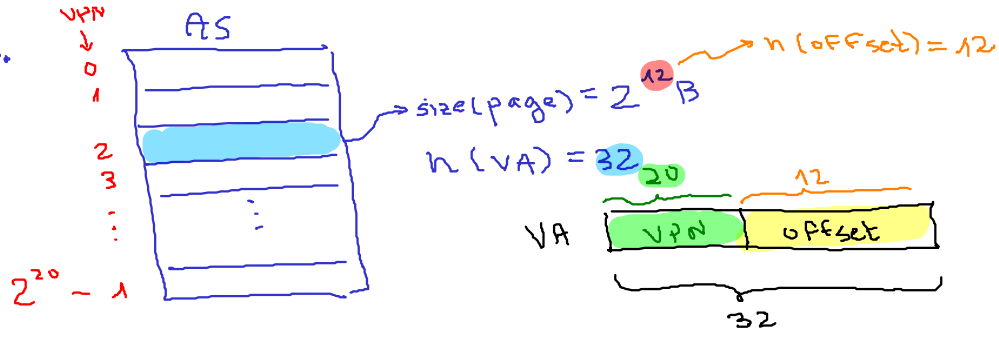
Suponga que se tiene una memoria virtual (Address Space) con las siguientes características:

- 32 bits de direcciones. → n(VA) = 32 bits
- Páginas de 4KB → size(page) = 4KB = 4x1024B = $2^2 2^{10}$ B = $2^{12}$ B
- Asuma que cada PTE tiene un tamaño de 4B → size(PTE) = 4B

Responda las siguientes preguntas:

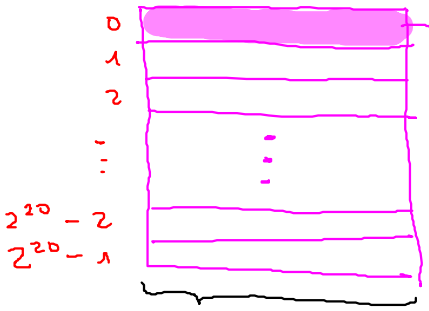
1. ¿Cual es el formato de las VA en este caso? → VA [VPN | offset] ?
2. ¿Cual es el tamaño de la tabla de página (PT)? → size(PT) = ?

Solución: 1.



$$pages = \frac{size(AS)}{size(page)} = \frac{2^{32}}{2^{12}} = 2^{20} \text{ pages.}$$

## 2. size (PT)




$nPTE = pages = 2^{20}$

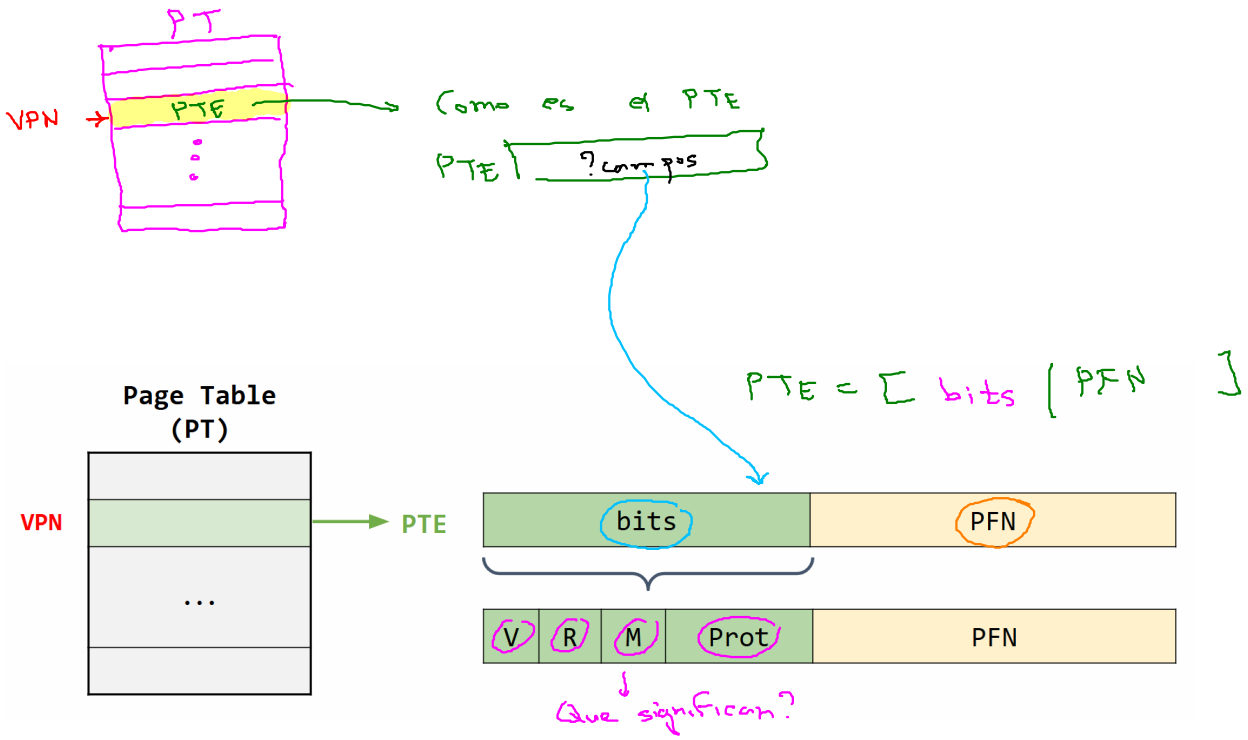
$size(PT) = [size(PTE)] \times nPTE = (4)(2^{20}) =$ **4 MB**

Que pasa si tengo n procesos?

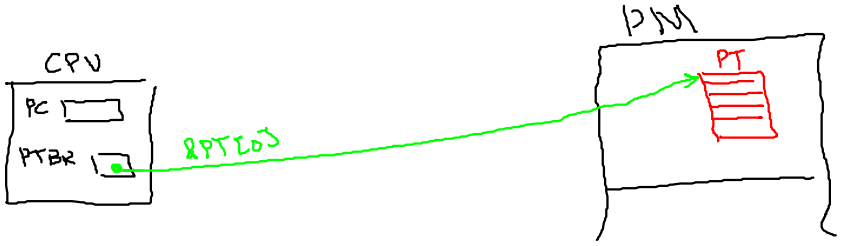
$size(All\ PTs) = n \times size(PT)$ $\xrightarrow{n=100}$ size [ All PT) = 100 (4M) = 400 MB

Mucho.

## 4. Como es una PT?




## 5. Registro PTBR



# 6. Protocolo de traducción de direcciones.

```
// Extract the VPN from the virtual address
VPN = (VirtualAddress & VPN_MASK) >> SHIFT

// Form the address of the page-table entry (PTE)
PTEAddr = PTBR + (VPN * sizeof(PTE))

// Fetch the PTE
PTE = AccessMemory(PTEAddr)

// Check if process can access the page
if (PTE.Valid == False)
    RaiseException(SEGMENTATION_FAULT)
else if (CanAccess(PTE.ProtectBits) == False)
    RaiseException(PROTECTION_FAULT)
else
    // Access is OK: form physical address and fetch it
    offset = VirtualAddress & OFFSET_MASK
    PhysAddr = (PTE.PFN << PFN_SHIFT) | offset
    Register = AccessMemory(PhysAddr)
```