

[SYS1] Systèmes d'exploitation (2)

Par Rhaeven, relecture par Find3r

Espace d'adressage

En demandant une adresse, l'espace alloué est constitué de zones de tailles fixes (4k)

Controlleur mémoire + appareils definis un espace d'adressage

- Espace d'adressage Physique de la machine :

4G	
	ROM
	Dev
	RAM
	Video
640k	RAM
0	

Mémoire vidéo : quand on y écrit, un device display le contenu sur un écran

Adressage avec segments

Segment : Quand on décrit une zone, on décrit une **base** (l'adresse) et une **limite** (taille, offset de la dernière adresse valide = taille - 1)

Dans le processeur, un registre permet de stocker le segment courant

Modele bien pourri (prof)

CODE
^
Data

On écrit les datas en avançant via le segment dans une zone linéaire, on va donc finir par écrire sur une autre zone (le data va finir par écrire sur la zone du code)

Virtualisation de mémoire

Adressage virtuel	Adressage physique
	P1
V1	
V2	
	P2

Ou V1 pointe vers P1 et V2 pointe vers P2

On appelle ça aussi la **pagination** :

- On découpe notre espace d'adressage en zone (des *pages*) de globalement 4Kb
- **Adresse de la page** encodée sur 20bits

Le processeur fait le déréférencement en passant par la table de traduction (on lui donne l'adresse virtuelle et elle va l'associer à l'adresse physique)

MMU : Memory Managment Unit (puce qui s'occupe de faire la traduction)

En plus de stocker l'adresse des adresses physiques, la table de traduction doit aussi stocker des **permissions**

@V	@Phy	flag

Comment encoder cette table ?

	4M
@Px0	@Phy 20b flags

Mémoire doit être contigue en mémoire physique > c'est relativement complexe
-> On utilise une structure arborescente = un **BTree**

- Node : fait la taille d'une page
- Pointe soit sur l'adresse de la page suivante soit sur de la data

4
p

p pointe vers

4
d

