

Gestion Mémoire : plan et illustrations

1 Multiprogrammation, objectifs et conséquences

1.1 Contexte

-
-

1.2 Partage du temps

-
-

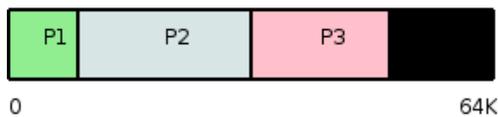
1.3 Cohabitation en mémoire

Espace mémoire



Où mettre les programmes ?

Espace mémoire

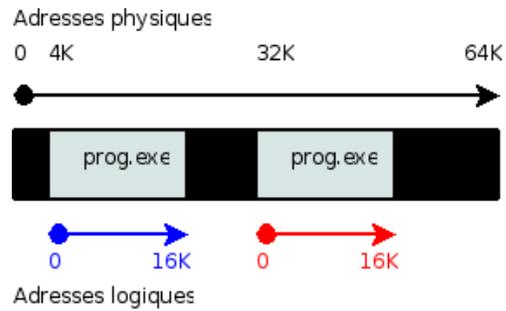


à la suite les uns des autres ?

2 Adresses logiques

-
-

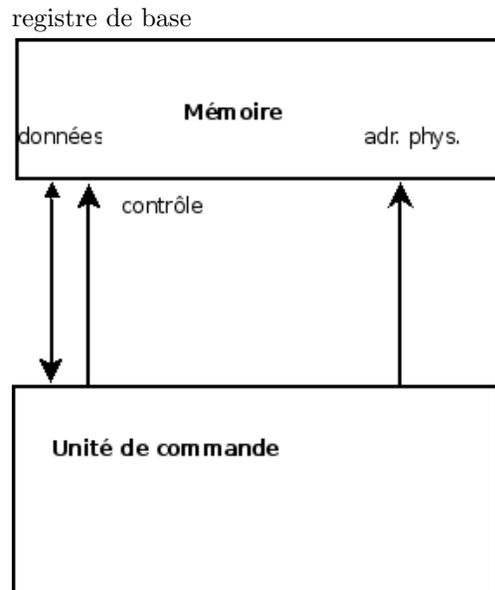
2.1 Indépendance code/position physique



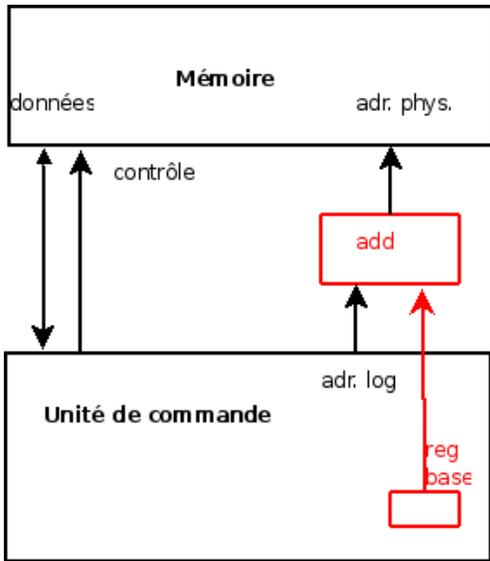
Chaque processus a un espace d'adressage logique

Adresse physique = adresse de base + adresse logique

2.2 Modification du matériel

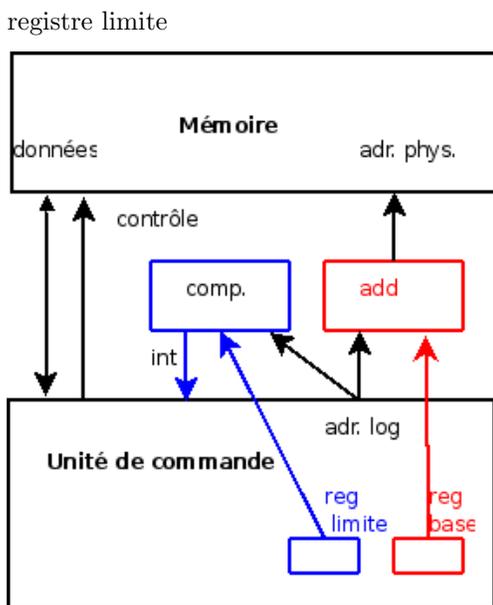


(architecture initiale)



avec registre de base

2.3 Protection inter-processus



avec registre limite

3 Gestion de l'allocation contiguë

3.1 Problème : la fragmentation

3.2 Exemple de scénario

Espace mémoire



(0) 3 processus sont présents, P2 se termine ...

Espace mémoire



(1) allocation de P4 ...

Espace mémoire



(3) allocation de P5 ...

Espace mémoire



(4) libération de P4 ...

Espace mémoire



(5) l'espace libre est fragmenté

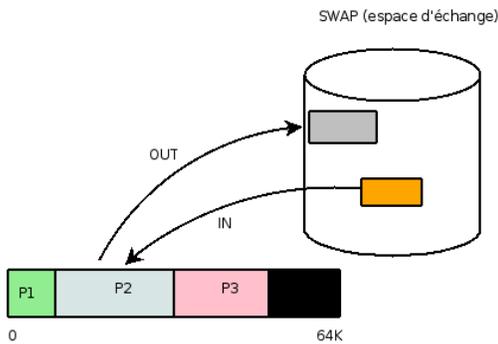
3.3 Solutions

-
-

4 Swapping

- but ?
-

4.1 idée



Echanges mémoire ↔ disque

4.2 Avantages

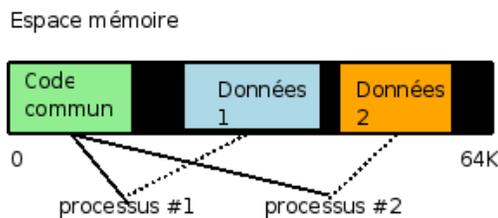
-
-

4.3 Remarque

-
-

5 Segmentation

5.1 Idée



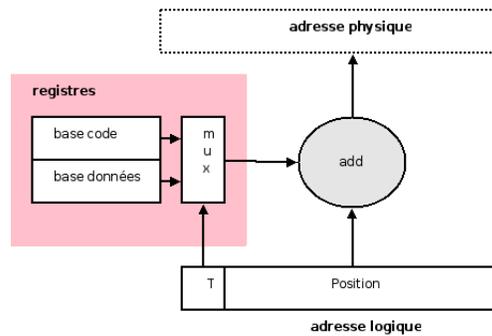
Deux processus exécutent le même code

5.2 Réalisation

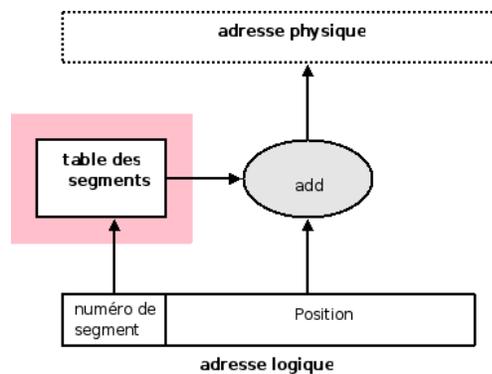
Bits d'une adresse



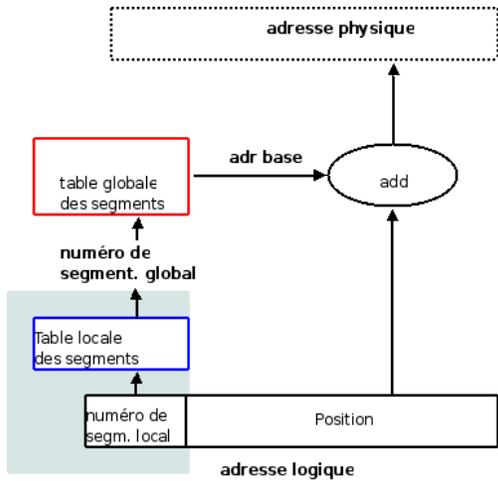
T=0 : position dans le segment de code
T=1 : position dans le segment de données



un multiplexeur sélectionne le registre de base à utiliser



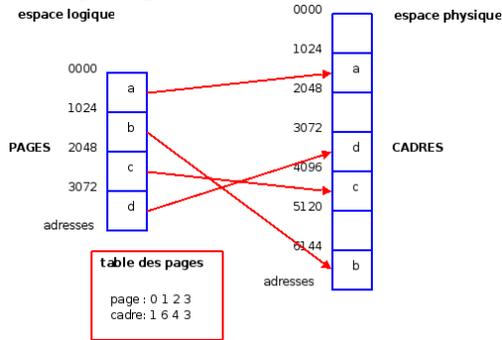
utilisation d'une table des segments



Génération d'adresses, TLS + TGS

6 Espace mémoire paginé

But ? principe ?

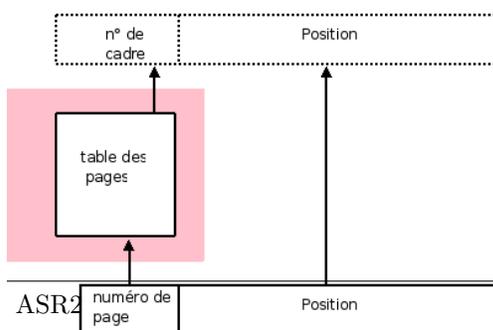


espace paginé

Bits d'une adresse

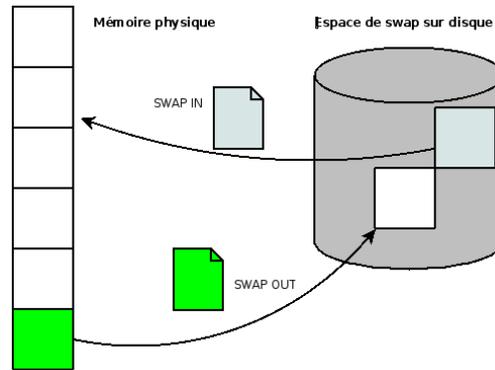


format d'adresse logique
adresse physique



génération d'adresse

7 Mémoire virtuelle paginée

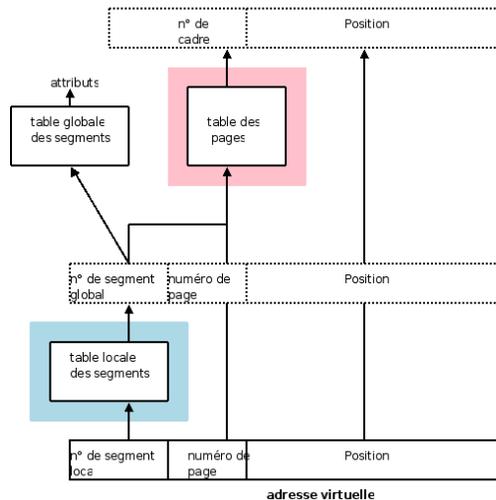


Swapping

8 Algorithmes de remplacement de page

9 Mémoire virtuelle segmentée-paginée

Utilisée depuis l'IBM 360/67 (1969)
adresse physique



génération d'adresse