La gestion de la mémoire

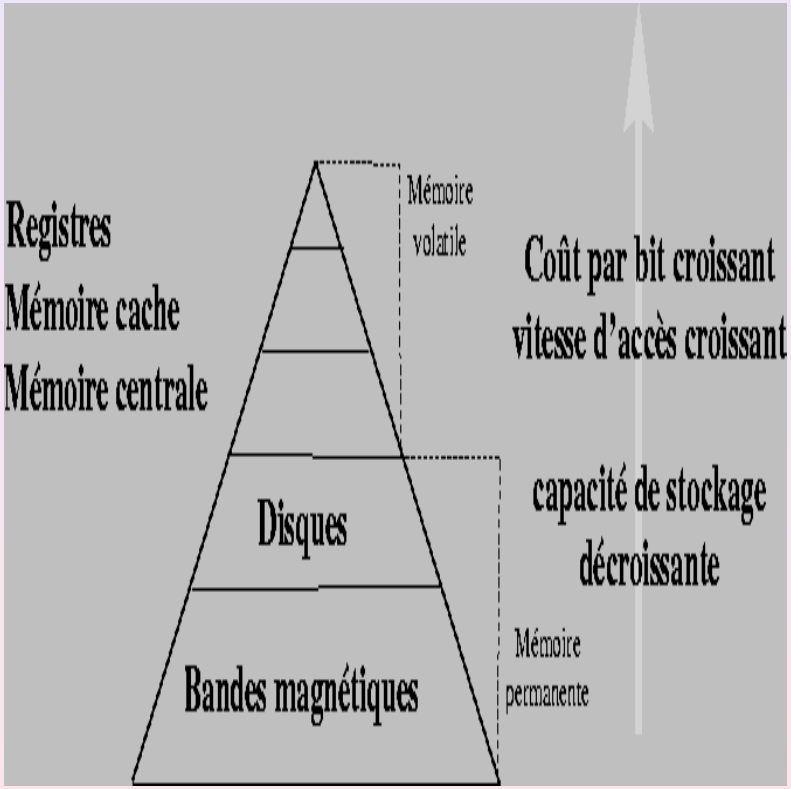
Luigi Santocanale

Laboratoire d'Informatique Fondamentale,
Centre de Mathématiques et Informatique,
39, rue Joliot-Curie - F-13453 Marseille

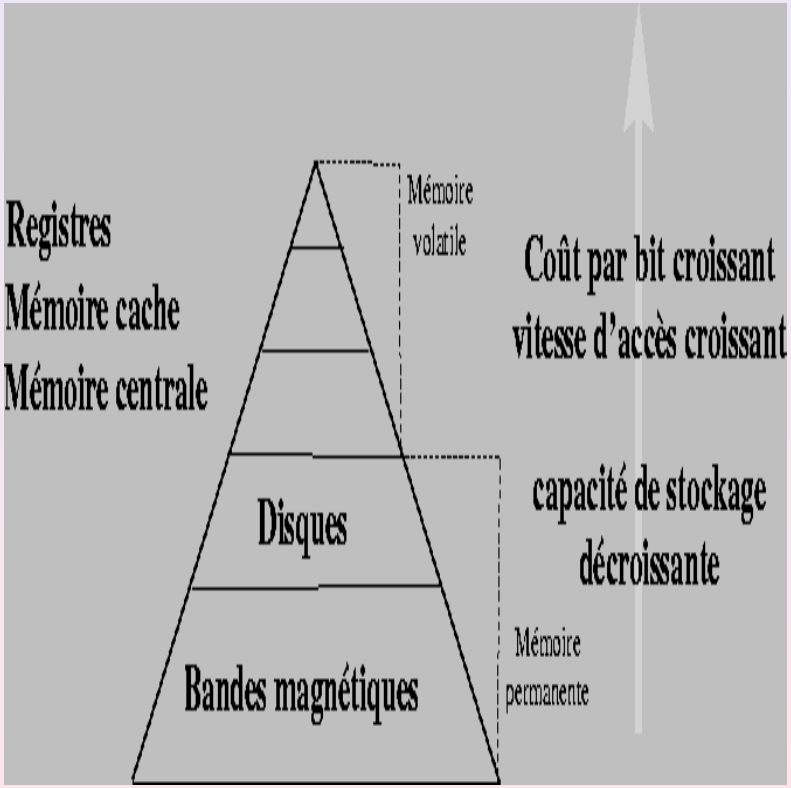
7 décembre 2005

Plan

Les mémoires



Les mémoires

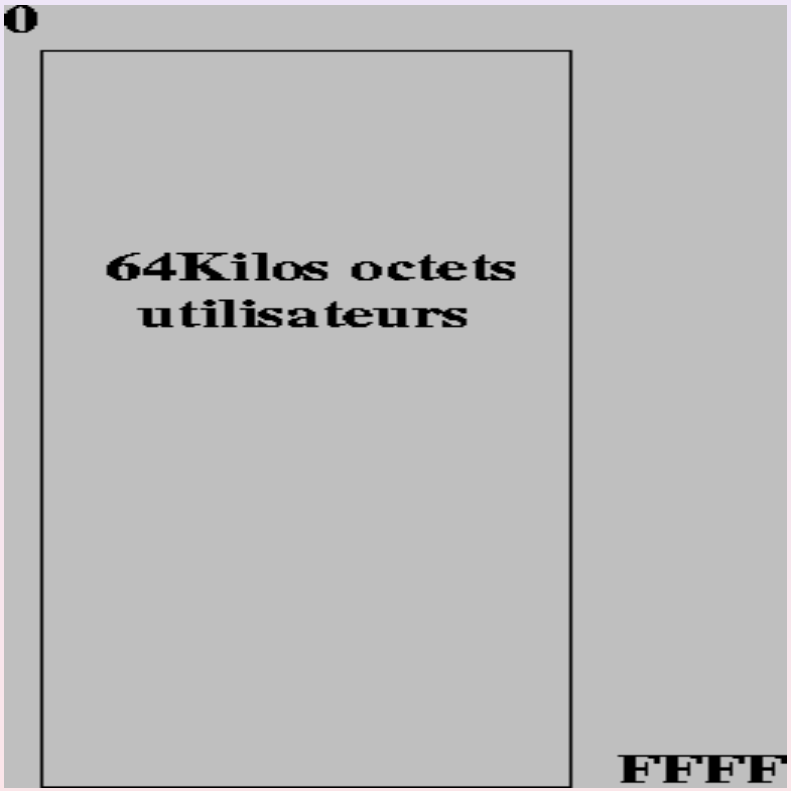


CARACTERISTIQUES DES TYPES DE MEMOIRES

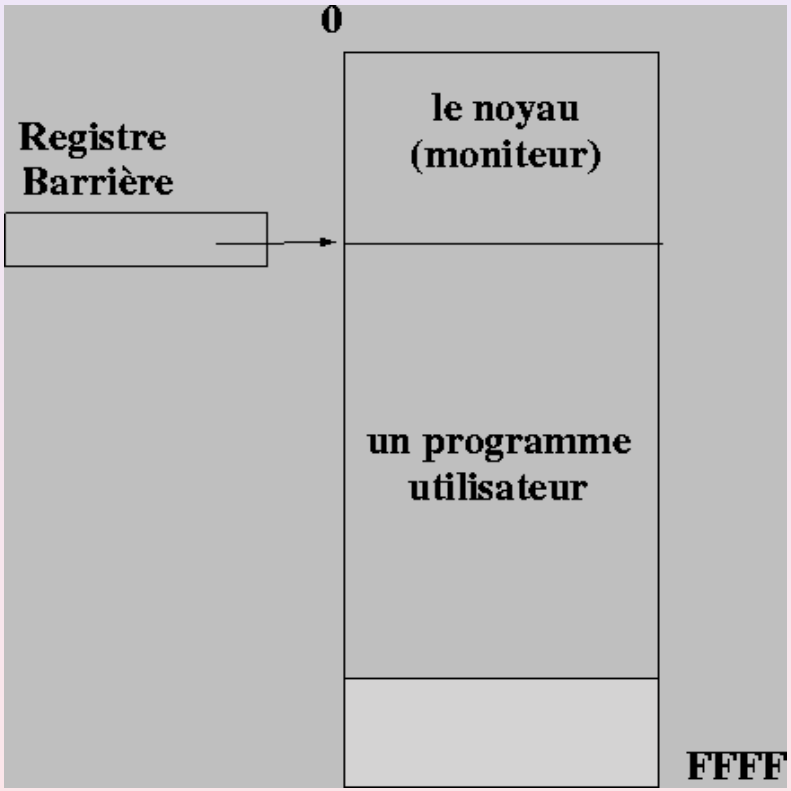
TYPE DE MEMOIRE	TAILLE (Octets)	TEMPS D'ACCES (secondes)	COUT RELATIF PAR BIT
CACHE	10^3-10^4	10^{-8}	10
MEMOIRE CENTRALE	10^6-10^7	10^{-7}	1
DISQUE	10^8-10^9	10^{-3}-10^{-2}	10^{-2}-10^{-3}
BANDE	10^8-10^9	10-10^2	10^{-4}

Plan

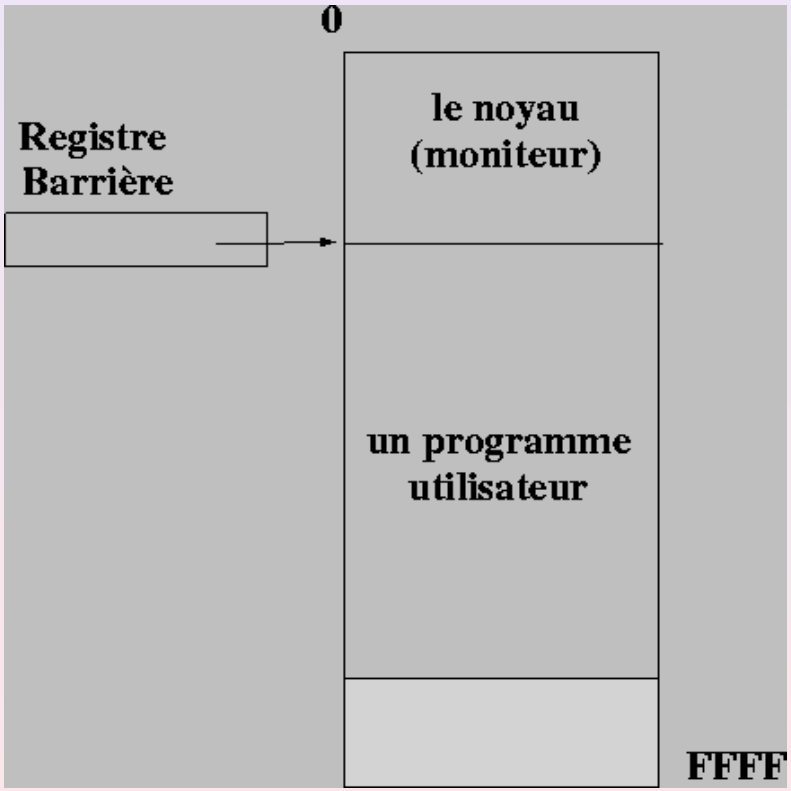
Protection du noyau : le registre barrière

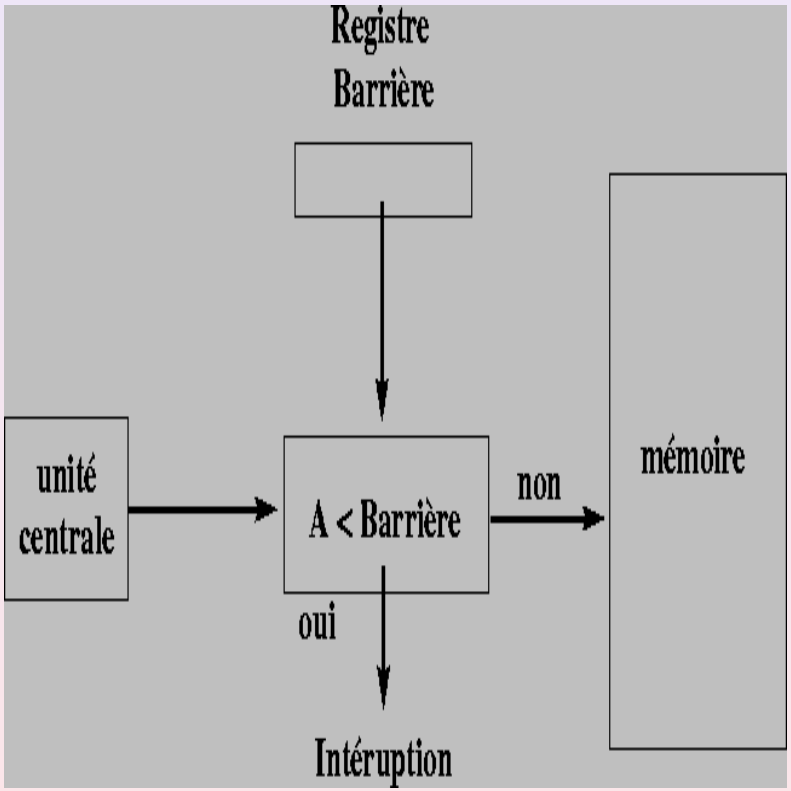


Protection du noyau : le registre barrière

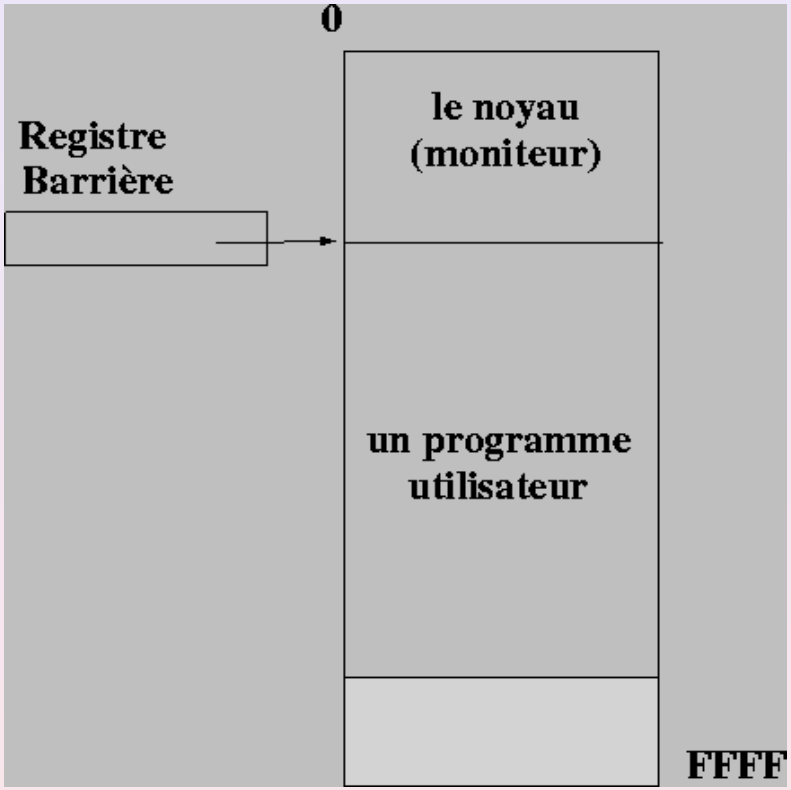


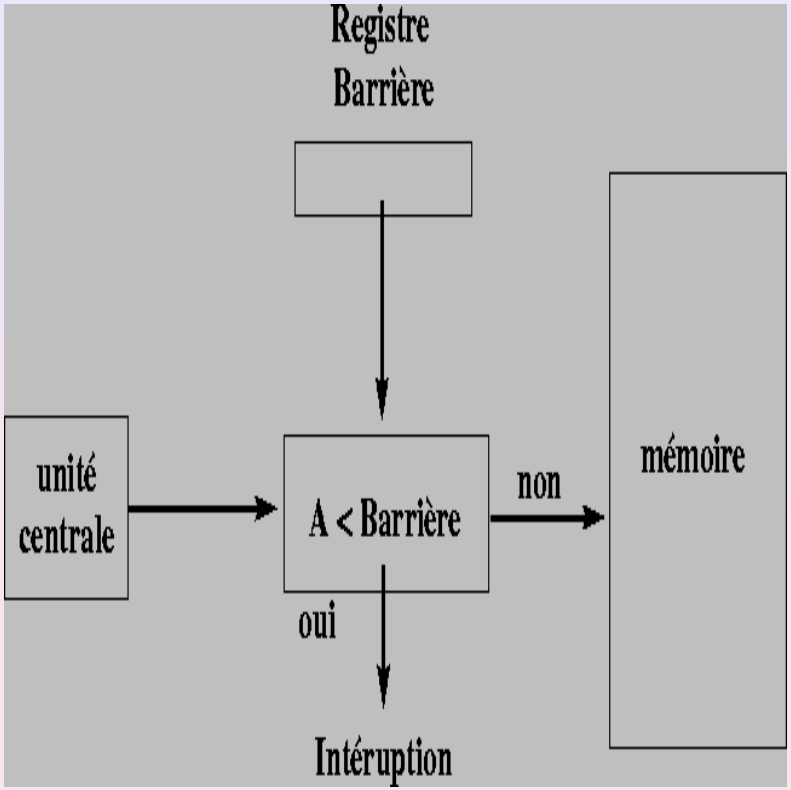
Protection du noyau : le registre barrière





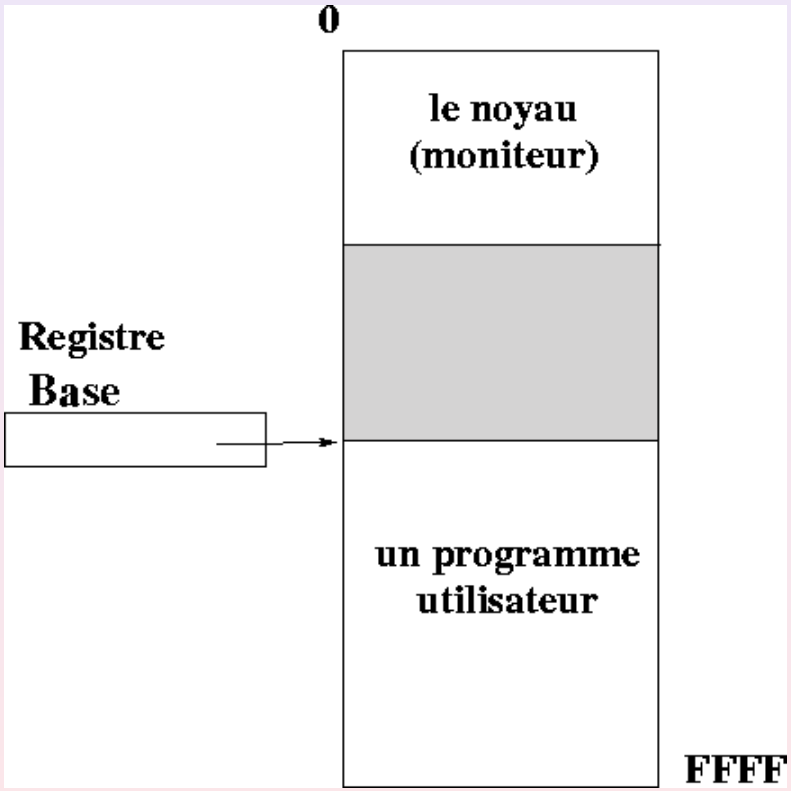
Protection du noyau : le registre barrière



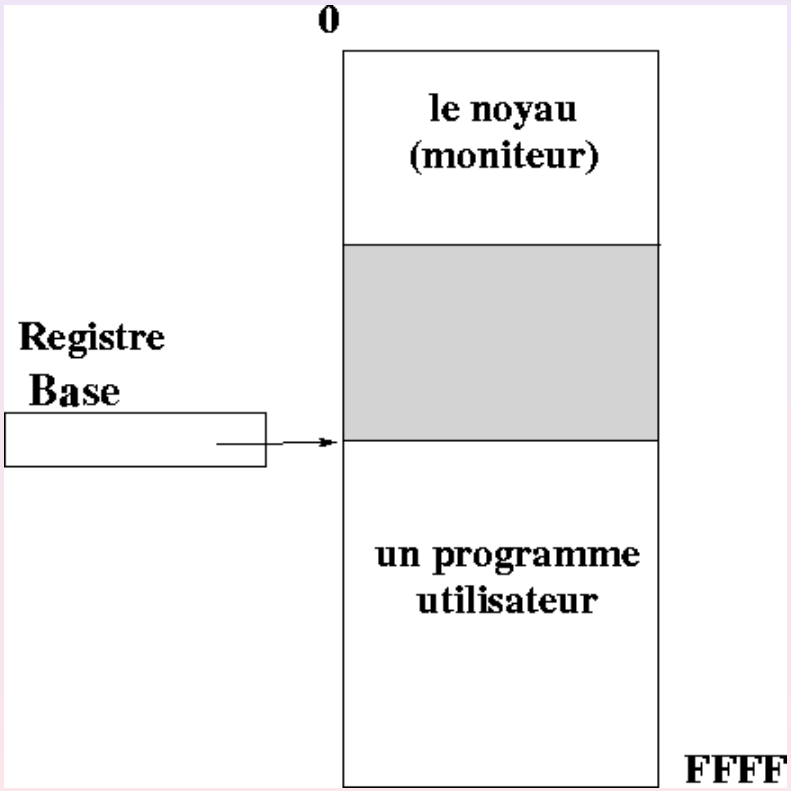


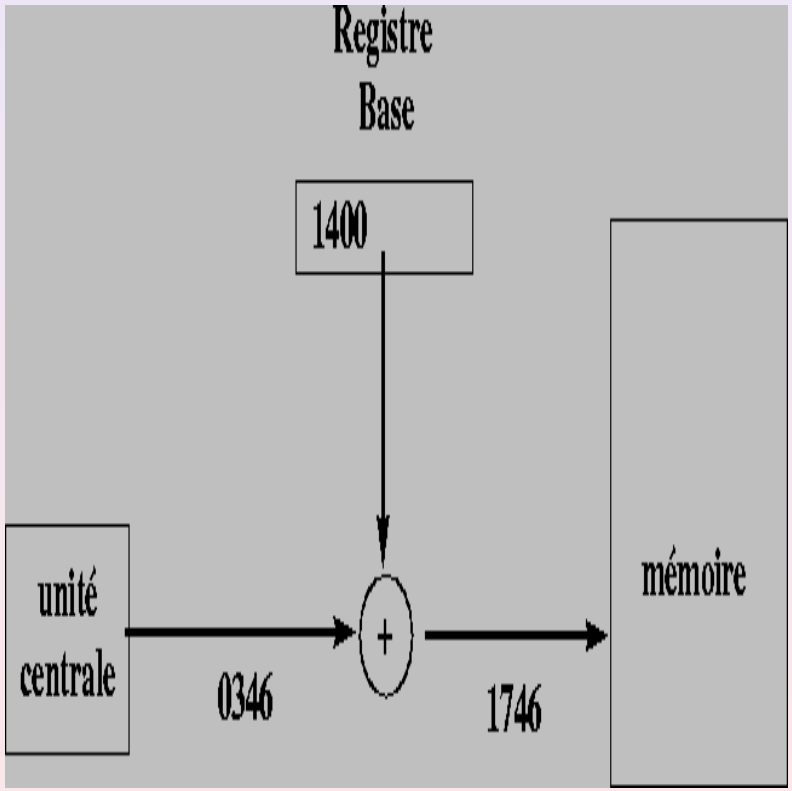

```
si ( adresse <= Barriere )
    lever exception
sinon
    utiliser adresse
```

Protection du noyau : le registre base

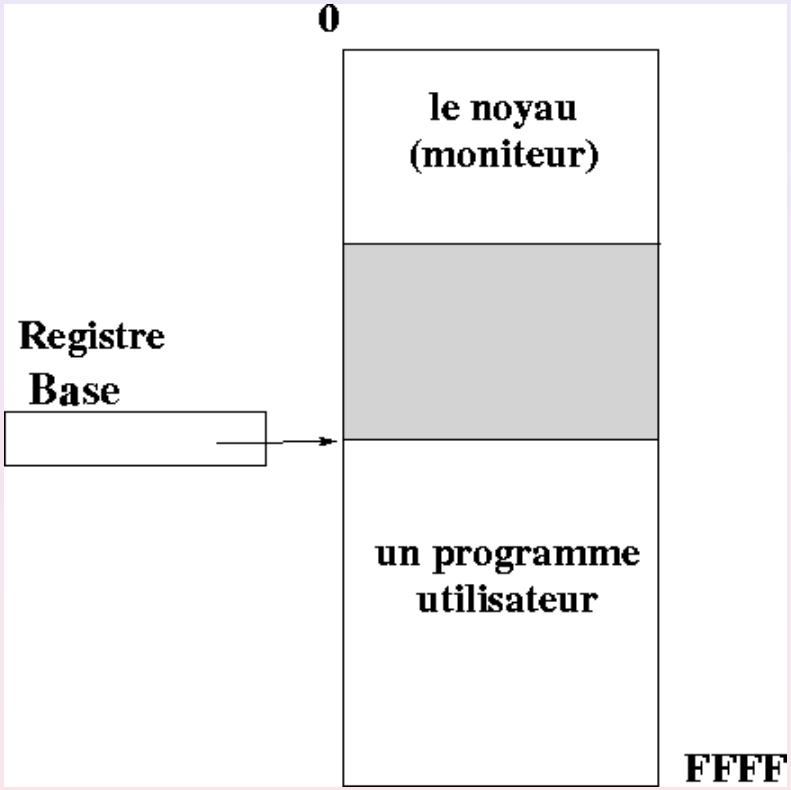


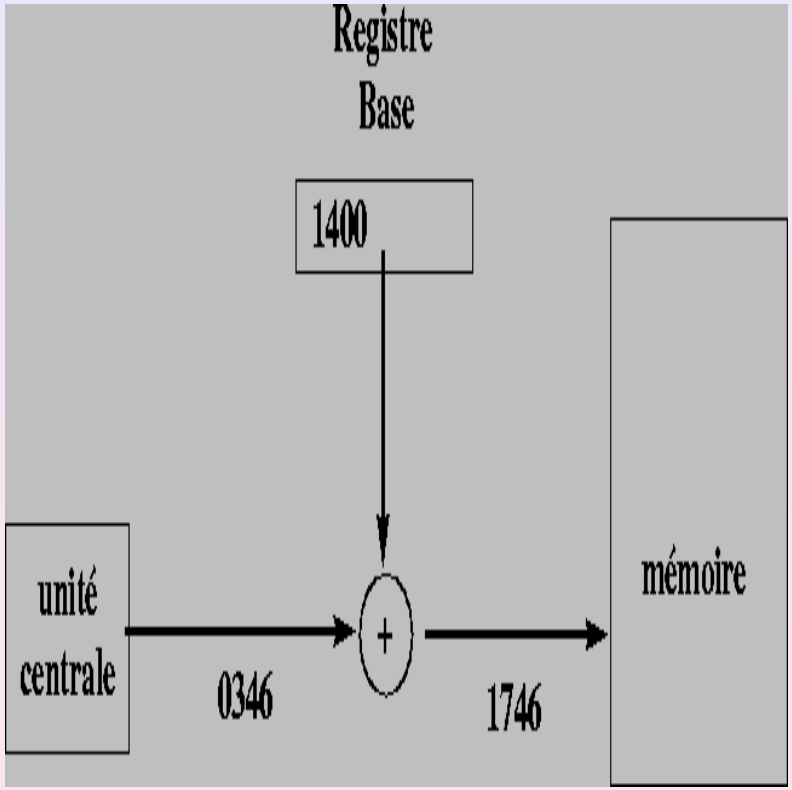
Protection du noyau : le registre base





Protection du noyau : le registre base






```
adressephysique =
    adresselogique + Base
```

Adresses logiques vs. adresses physiques

Adresses logiques :

- privé au le programme, utilisé par l'unité centrale.

Adresses physiques :

- accès à la mémoire, instructions de lecture/écriture de/à un registre,
- la MMU (« memory management unit ») transforme les adresses logiques en adresses physiques.

Adresses logiques vs. adresses physiques

Adresses logiques :

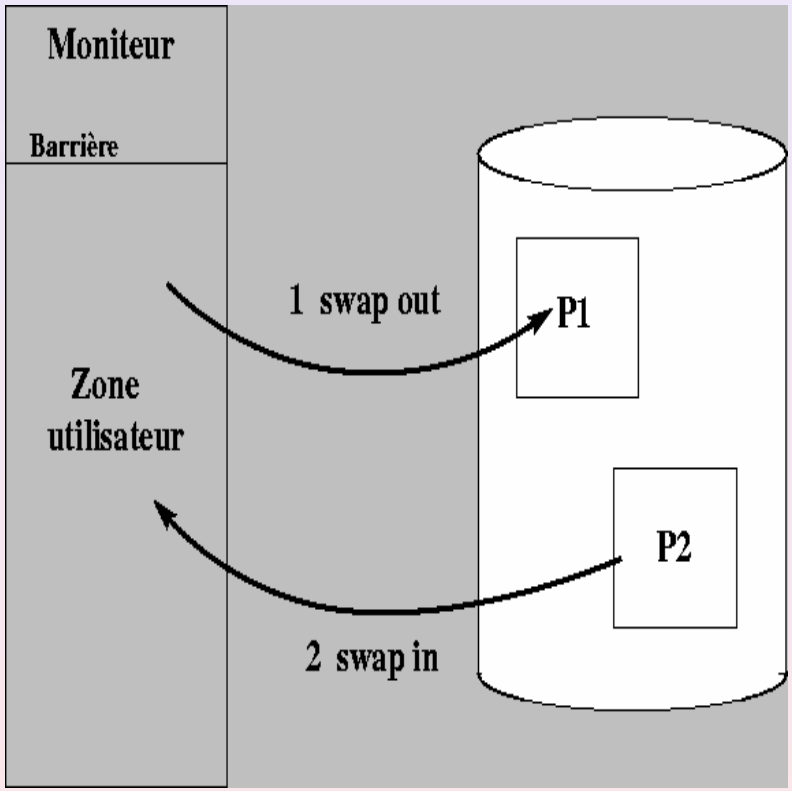
- privé au le programme, utilisé par l'unité centrale.

Adresses physiques :

- accès à la mémoire, instructions de lecture/écriture de/à un registre,
- la MMU (« memory management unit ») transforme les adresses logiques en adresses physiques.

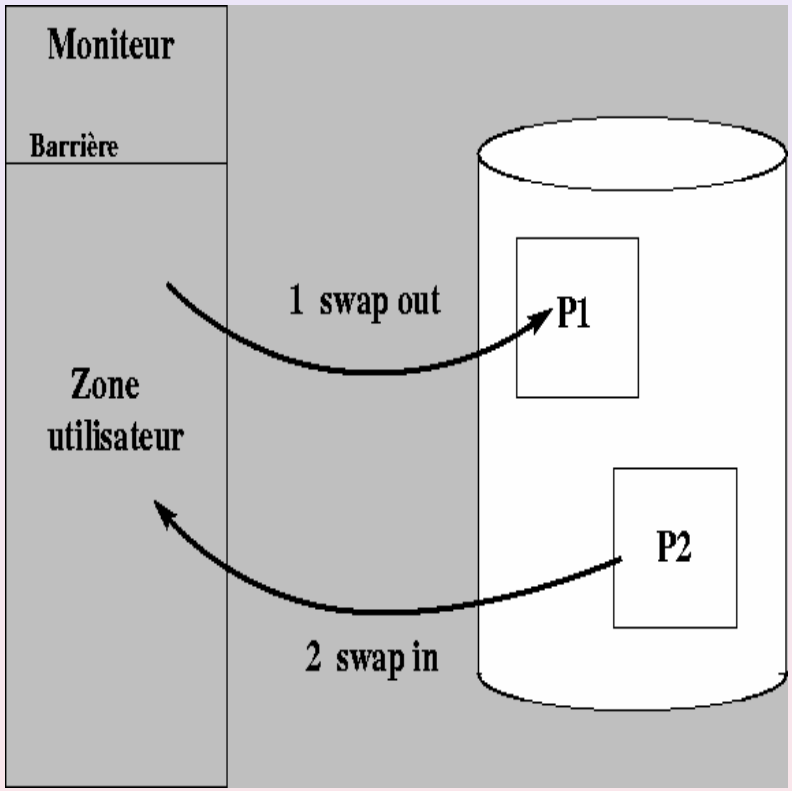
Plan

Le swap



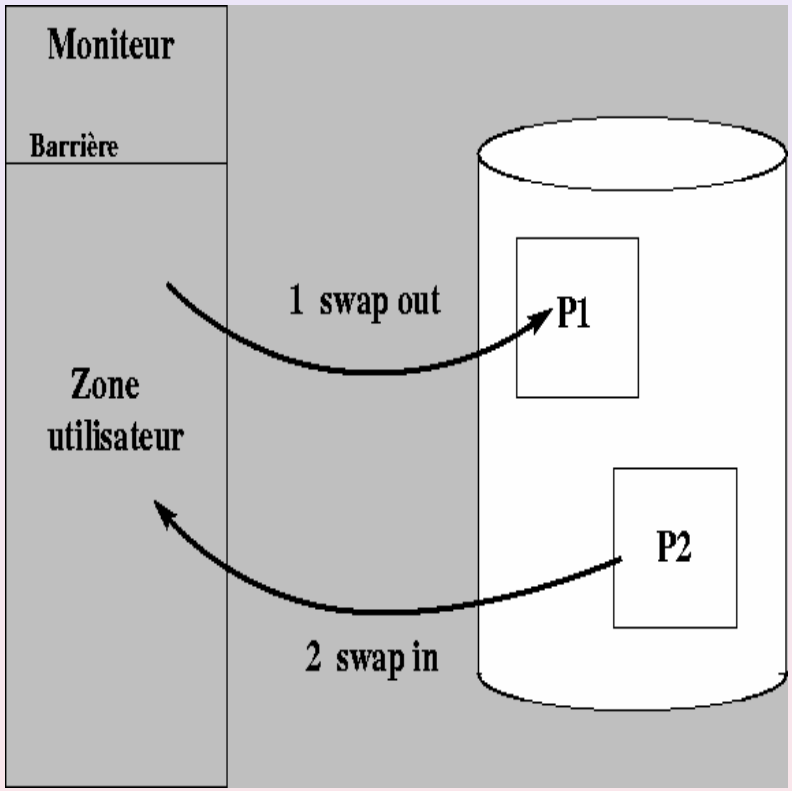
- Coût
- Taille des processus
- Contraintes sur les E/S :
  - ne pas swapper des processus en attente d'E/S
  - réaliser des buffers de E/S dans le noyau (voir UNIX).

Le swap



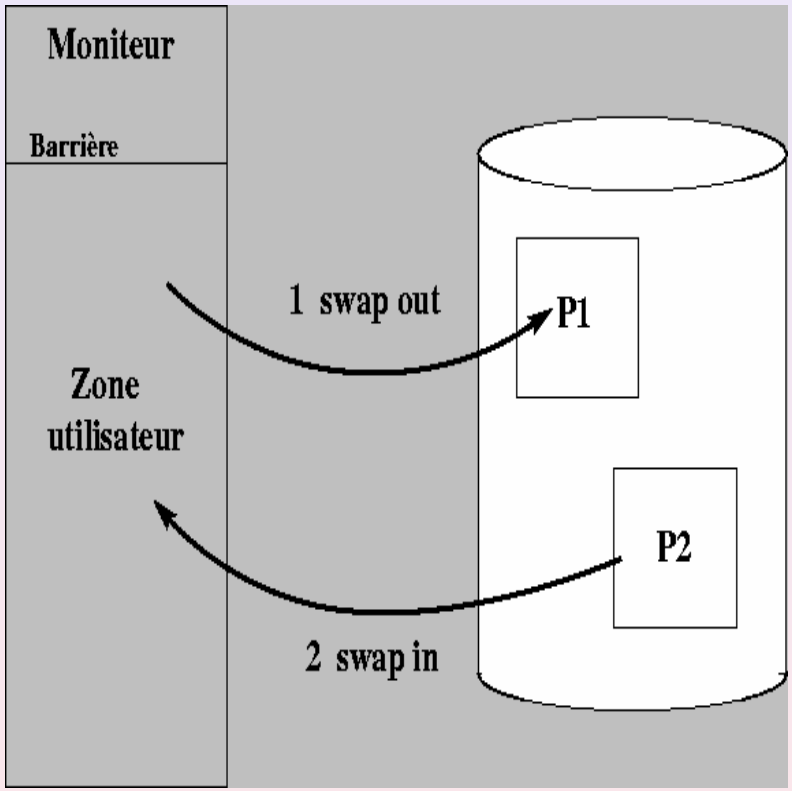
- Coût
- Taille des processus
- Contraintes sur le E/S :
  - ne pas swapper les processus en attente de E/S
  - réaliser des buffers de E/S dans le noyau (voir UNIX).

Le swap



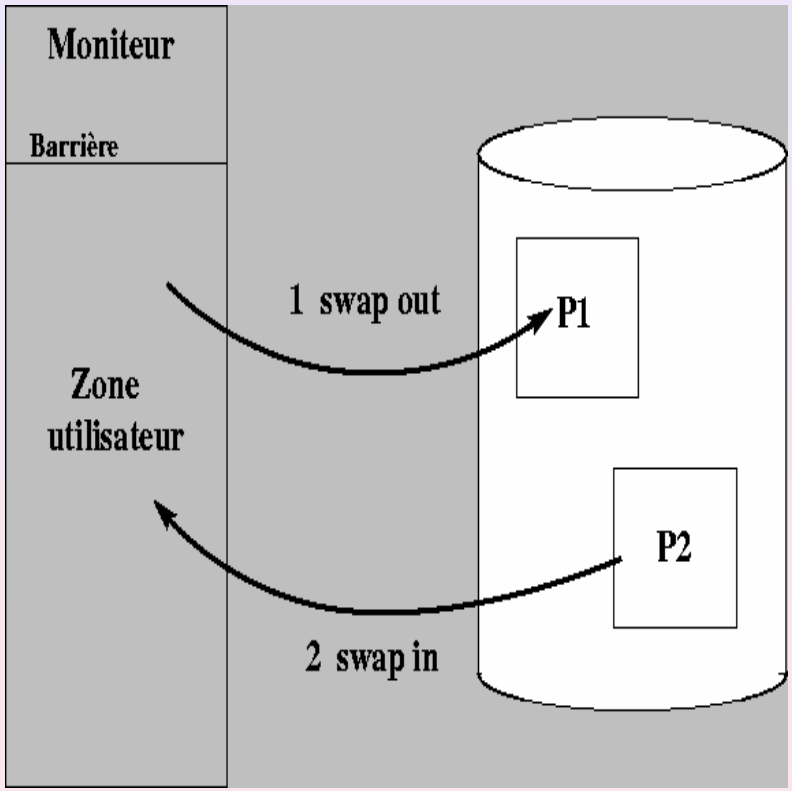
- Coût
- Taille des processus
- Contraintes sur le E/S :
  - ne pas swapper les processus en attante de E/S
  - réaliser des buffers de E/S dans le noyau (voir UNIX).

Le swap



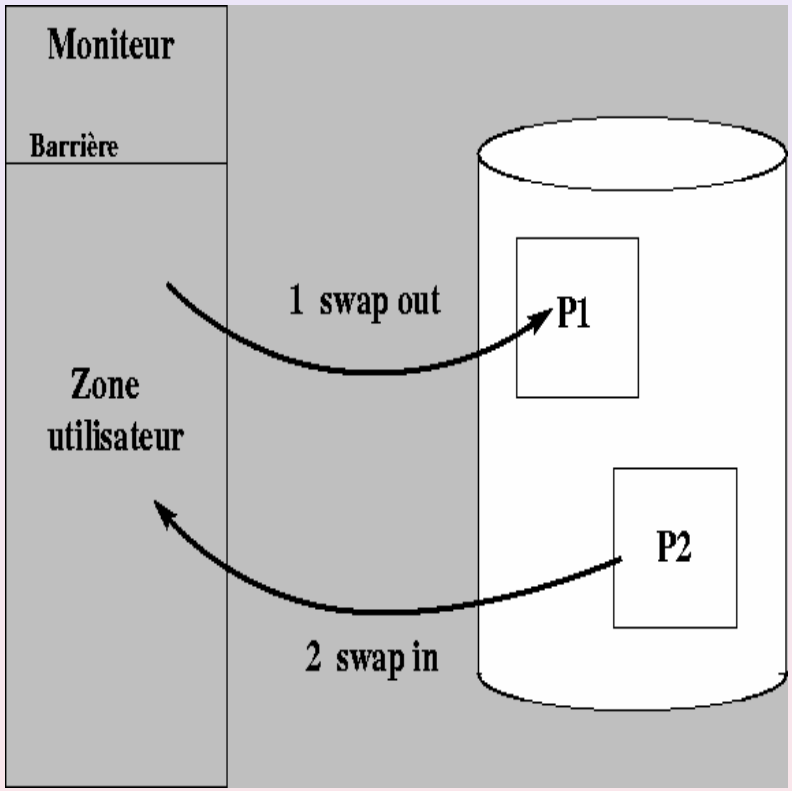
- Coût
- Taille des processus
- Contraintes sur le E/S :
  - ne pas swapper les processus en attante de E/S
  - réaliser des buffers de E/S dans le noyau (voir UNIX).

Le swap



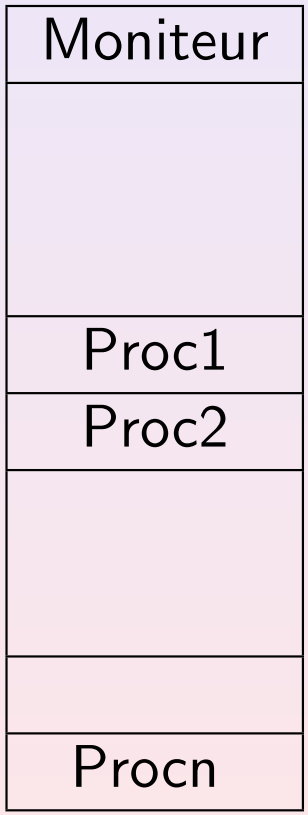
- Coût
- Taille des processus
- Contraintes sur le E/S :
  - ne pas swapper les processus en attante de E/S
  - réaliser des buffers de E/S dans le noyau (voir UNIX).

Le swap



- Coût
- Taille des processus
- Contraintes sur le E/S :
  - ne pas swapper les processus en attante de E/S
  - réaliser des buffers de E/S dans le noyau (voir UNIX).

Le partage de la mémoire entre processus

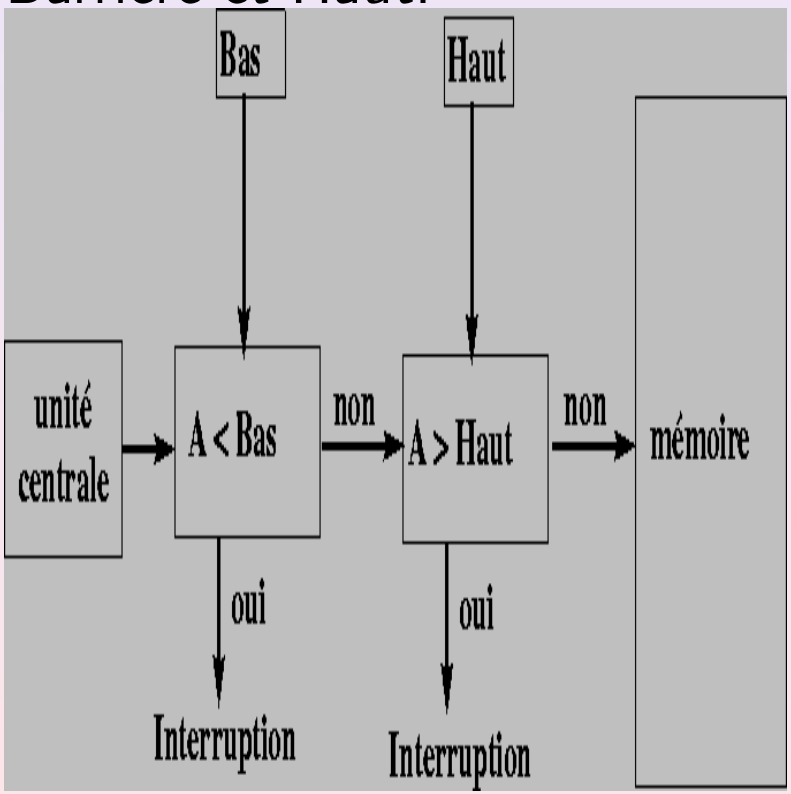


Problèmes :

- Protection entre processus
- Allocation de la mémoire aux processus (ordonnancement)

La protection entre processus : deux registres

Barrière et Haut:



Ordonnancement en mémoire

Comment choisir l'endroit où charger un nouveaux processus :

- First-fit : premier bloc suffisamment grand.
- Best-fit : plus petit bloc suffisamment grand.
- Worst-fit : le bloc qui nous laisse le plus grand bloc libre (le plus grand bloc).

Problème : la fragmentation.

Ordonnancement en mémoire

Comment choisir l'endroit où charger un nouveaux processus :

- First-fit : premier bloc suffisamment grand.
- Best-fit : plus petit bloc suffisamment grand.
- Worst-fit : le bloc qui nous laisse le plus grand bloc libre (le plus grand bloc).

Problème : la fragmentation.

Ordonnancement en mémoire

Comment choisir l'endroit où charger un nouveaux processus :

- First-fit : premier bloc suffisamment grand.
- Best-fit : plus petit bloc suffisamment grand.
- Worst-fit : le bloc qui nous laisse le plus grand bloc libre (le plus grand bloc).

Problème : la fragmentation.

Ordonnancement en mémoire

Comment choisir l'endroit où charger un nouveaux processus :

- First-fit : premier bloc suffisamment grand.
- Best-fit : plus petit bloc suffisamment grand.
- Worst-fit : le bloc qui nous laisse le plus grand bloc libre (le plus grand bloc).

Problème : la fragmentation.

Ordonnancement en mémoire

Comment choisir l'endroit où charger un nouveaux processus :

- First-fit : premier bloc suffisamment grand.
- Best-fit : plus petit bloc suffisamment grand.
- Worst-fit : le bloc qui nous laisse le plus grand bloc libre (le plus grand bloc).

Problème : la fragmentation.

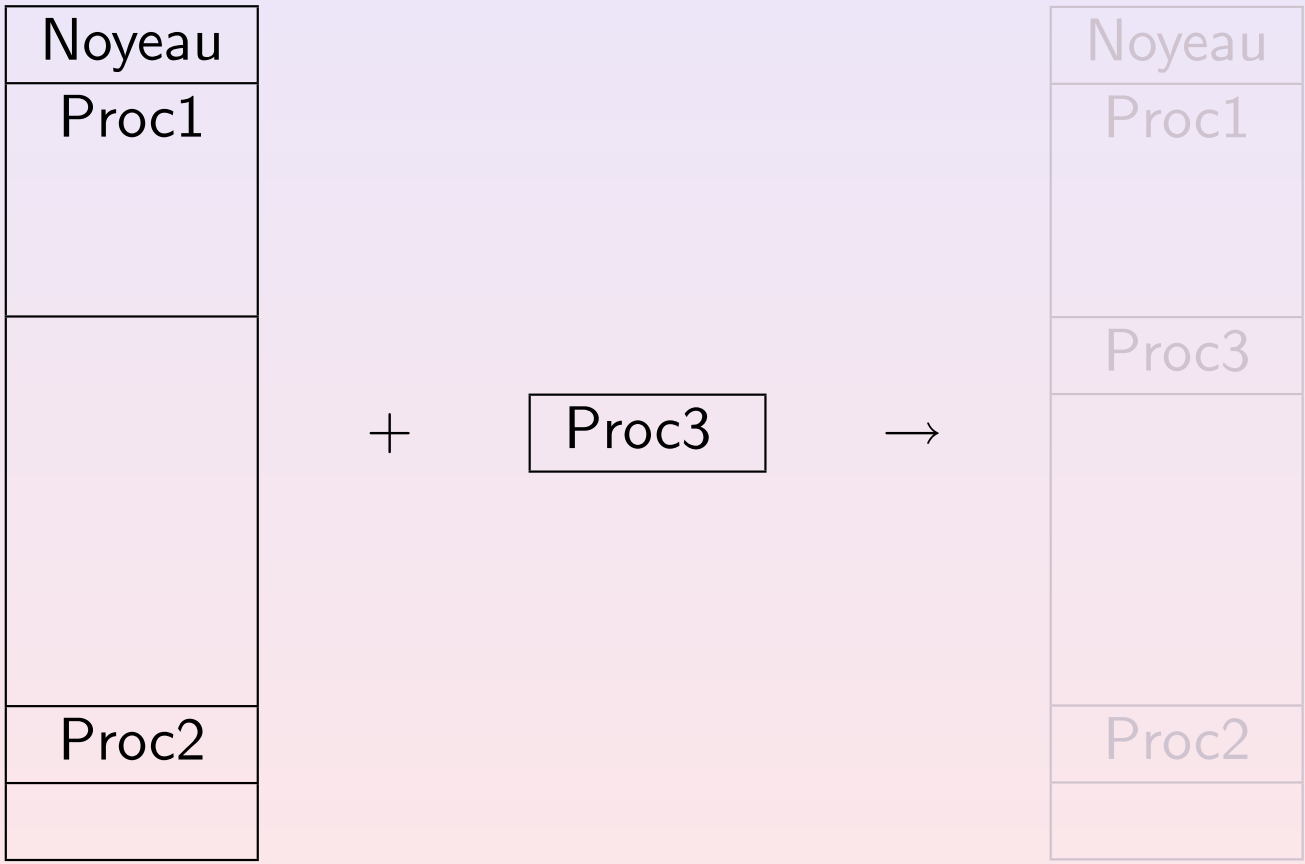
Ordonnancement en mémoire

Comment choisir l'endroit où charger un nouveaux processus :

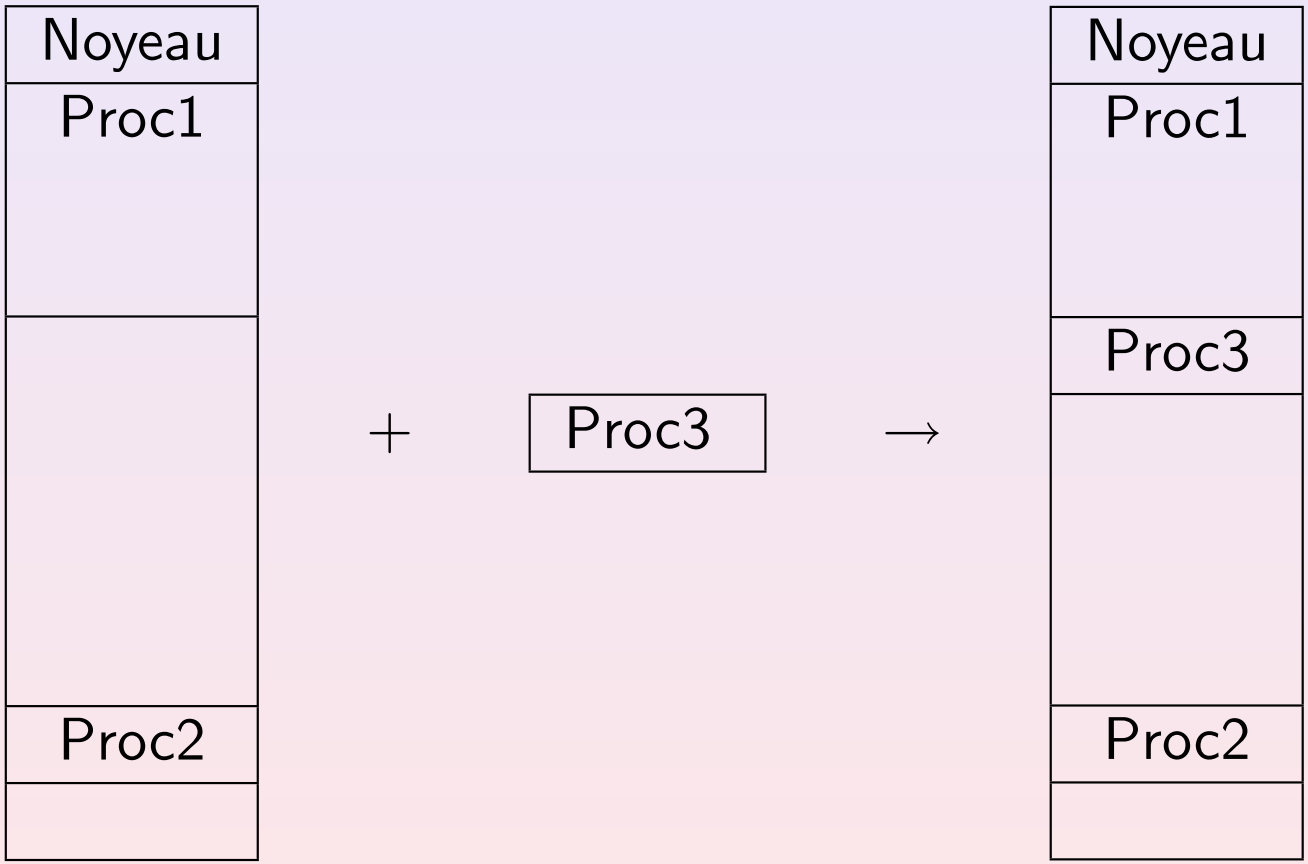
- First-fit : premier bloc suffisamment grand.
- Best-fit : plus petit bloc suffisamment grand.
- Worst-fit : le bloc qui nous laisse le plus grand bloc libre (le plus grand bloc).

Problème : la fragmentation.

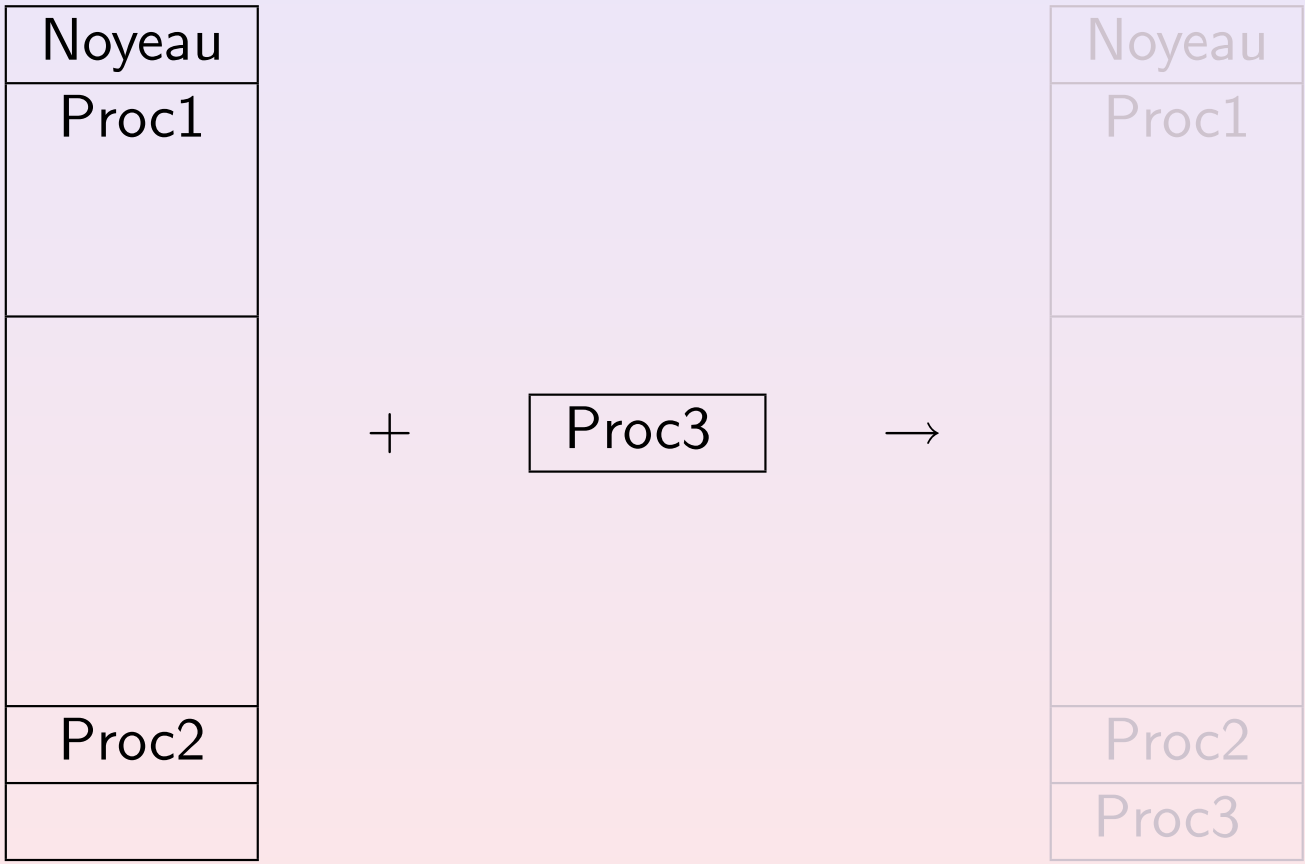
First fit



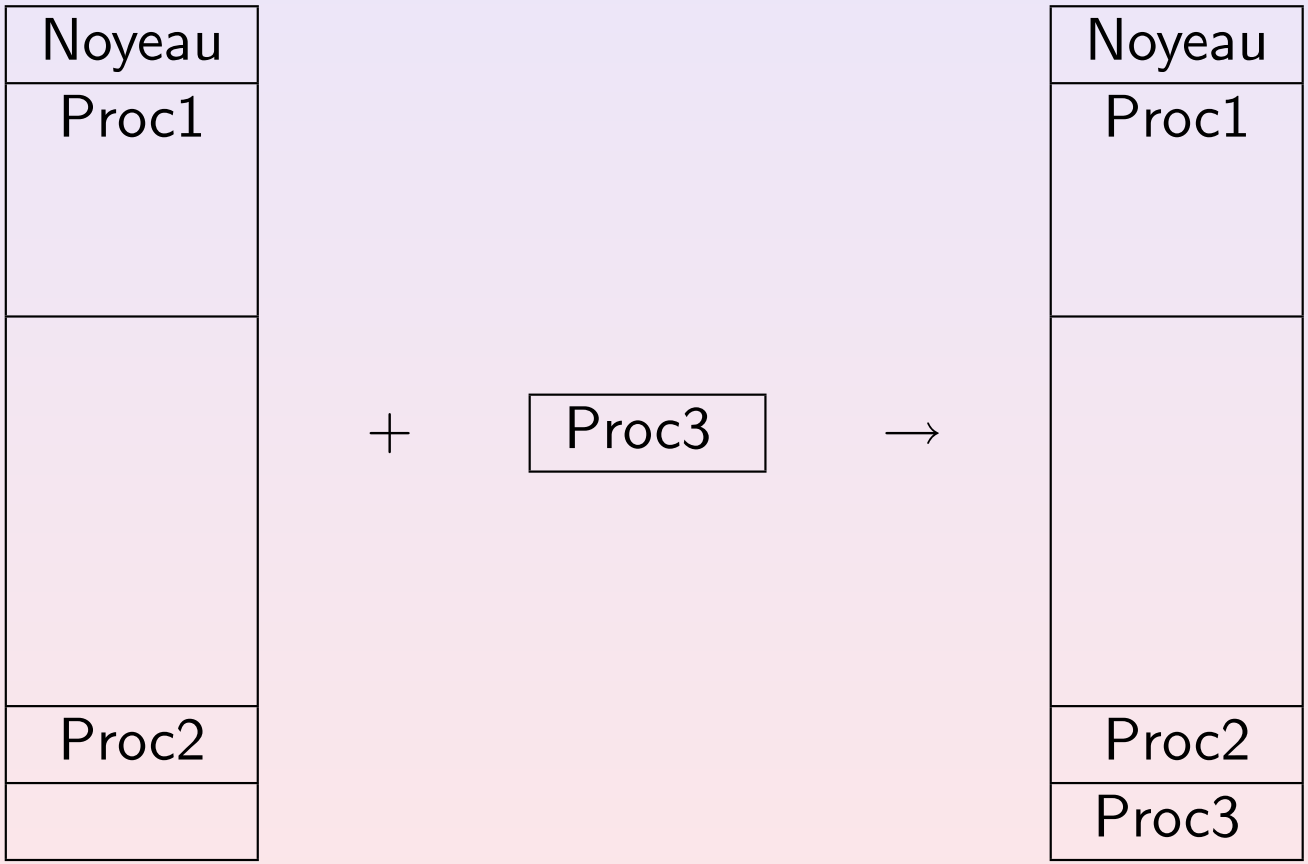
First fit



Best & worst fit

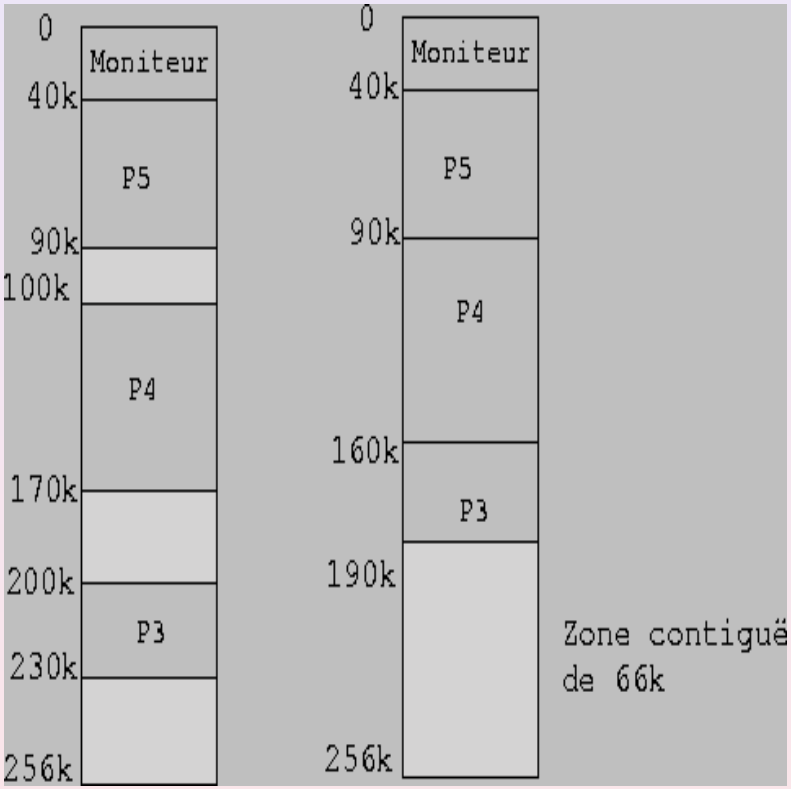


Best & worst fit



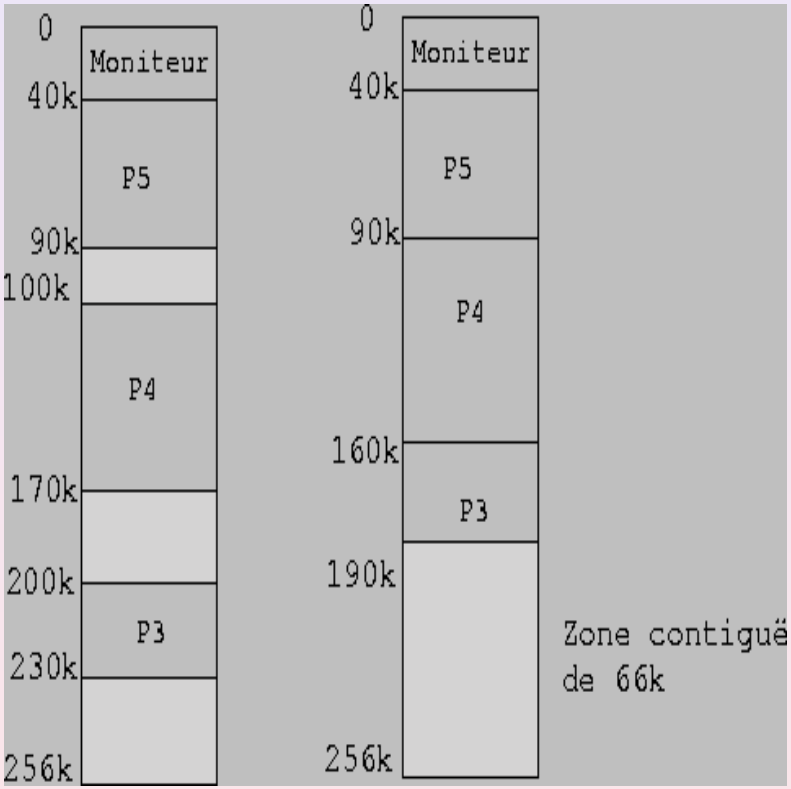
Le compactage

Solution à la fragmentation.

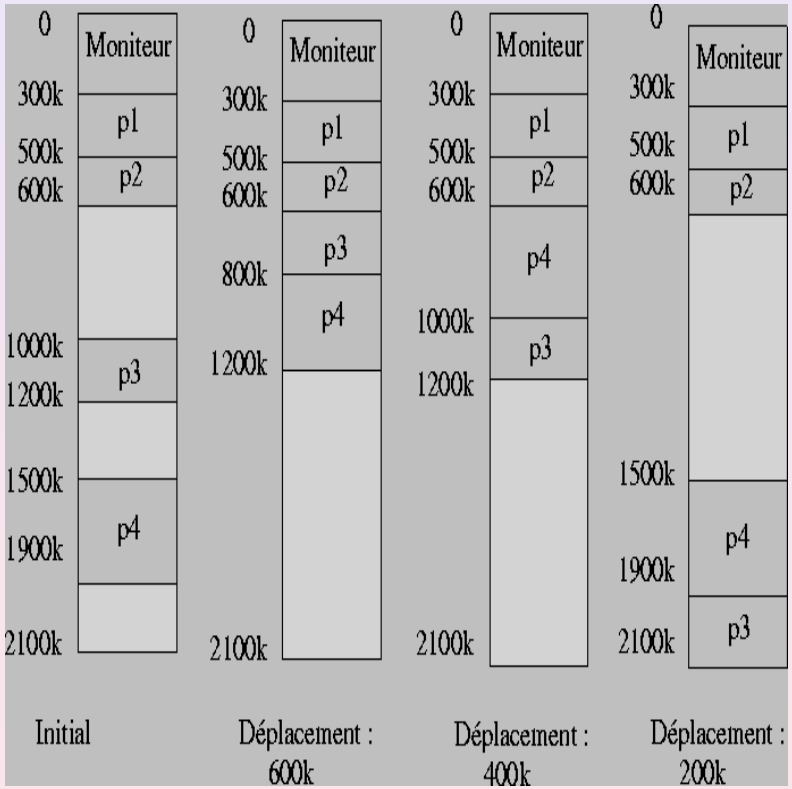


Le compactage

Solution à la fragmentation.



Exemples :



Plan

La pagination

But : solution à la fragmentation,
Voir : la gestion de l'espace sur disque, les bloques.

- La mémoire logique est découpée en pages.
- Une adresse logique est découpée en une couple :
  (page, déplacement)

Si :
A = adresse logique, T = taille de page
alors
page = A/T, déplacement = $A\%T$

La pagination

But : solution à la fragmentation,
Voir : la gestion de l'espace sur disque, les bloques.

- La mémoire logique est découpée en pages.
- Une adresse logique est découpée en une couple :

(page, déplacement)

Si

$A \equiv$ adresse logique, $T \equiv$ taille de page

alors

$page \equiv A/T$, $déplacement \equiv A\%T$

La pagination

But : solution à la fragmentation,
Voir : la gestion de l'espace sur disque, les bloques.

- La mémoire logique est découpée en pages.
- Une adresse logique est découpée en une couple :
  (page, déplacement)

Si

$$A = \text{adresse logique}, \qquad T = \text{taille de page}$$

alors

$$\text{page} = A/T, \qquad \text{déplacement} = A\%T$$

La pagination

But : solution à la fragmentation,
Voir : la gestion de l'espace sur disque, les bloques.

- La mémoire logique est découpée en pages.
- Une adresse logique est découpée en une couple :
  (page, déplacement)

Si

$$A = \text{adresse logique}, \qquad T = \text{taille de page}$$

alors

$$\text{page} = A/T, \qquad \text{déplacement} = A\%T$$

La pagination

But : solution à la fragmentation,
Voir : la gestion de l'espace sur disque, les bloques.

- La mémoire logique est découpée en pages.
- Une adresse logique est découpée en une couple :

  (page, déplacement)

  Si

  $$A = \text{adresse logique}, \qquad T = \text{taille de page}$$

  alors

  $$\text{page} = A/T, \qquad \text{déplacement} = A\%T$$

La pagination (II)

- La mémoire physique est découpée en cadres de page (même taille d'une page).
- Adresses physique est un couple (f, d), cadre de pages et déplacement.
- A chaque page logique p peut correspondre un cadre de page $f = f(p)$.
- Cette correspondance est maintenue dans la table des pages.
- La MMU (memory management unit) calcule un adresse physique à partir d'un adresse logique selon la formule

$$phys(p, d) = (f(p), d).$$

Gestion de la multiprogrammation :

- Le PTBR, registre de base de la table de pages, « page-table base register ».

La pagination (II)

- La mémoire physique est découpée en cadres de page (même taille d'une page).
- Adresses physique est un couple (f, d), cadre de pages et déplacement.
- A chaque page logique p peut correspondre un cadre de page $f = f(p)$.
- Cette correspondance est maintenue dans la table des pages.
- La MMU (memory management unit) calcule un adresse physique à partir d'un adresse logique selon la formule

$$phys(p, d) = (f(p), d)\,.$$

Gestion de la multiprogrammation :

- Le PTBR, registre de base de la table de pages, « page-table base register ».

La pagination (II)

- La mémoire physique est découpée en cadres de page (même taille d'une page).
- Adresses physique est un couple (f, d), cadre de pages et déplacement.
- A chaque page logique p peut correspondre un cadre de page $f = f(p)$.
- Cette correspondance est maintenue dans la table des pages.
- La MMU (memory management unit) calcule un adresse physique à partir d'un adresse logique selon la formule

$$phys(p, d) = (f(p), d)\,.$$

Gestion de la multiprogrammation :

- Le PTBR, registre de base de la table de pages, « page-table base register ».

La pagination (II)

- La mémoire physique est découpée en cadres de page (même taille d'une page).
- Adresses physique est un couple (f, d), cadre de pages et déplacement.
- A chaque page logique p peut correspondre un cadre de page $f = f(p)$.
- Cette correspondance est maintenue dans la table des pages.
- La MMU (memory management unit) calcule un adresse physique à partir d'un adresse logique selon la formule

$$phys(p, d) = (f(p), d)\,.$$

Gestion de la multiprogrammation :

- Le PTBR, registre de base de la table de pages, « page-table base register ».

La pagination (II)

- La mémoire physique est découpée en cadres de page (même taille d'une page).
- Adresses physique est un couple (f, d), cadre de pages et déplacement.
- A chaque page logique p peut correspondre un cadre de page $f = f(p)$.
- Cette correspondance est maintenue dans la table des pages.
- La MMU (memory management unit) calcule un adresse physique à partir d'un adresse logique selon la formule

$$phys(p, d) = (f(p), d)\,.$$

Gestion de la multiprogrammation :

- Le PTBR, registre de base de la table de pages, « page-table base register ».

La pagination (II)

- La mémoire physique est découpée en cadres de page (même taille d'une page).
- Adresses physique est un couple (f, d), cadre de pages et déplacement.
- A chaque page logique p peut correspondre un cadre de page $f = f(p)$.
- Cette correspondance est maintenue dans la table des pages.
- La MMU (memory management unit) calcule un adresse physique à partir d'un adresse logique selon la formule

$$phys(p, d) = (f(p), d)\,.$$

Gestion de la multiprogrammation :

- Le PTBR, registre de base de la table de pages, « page-table base register ».

La pagination (II)

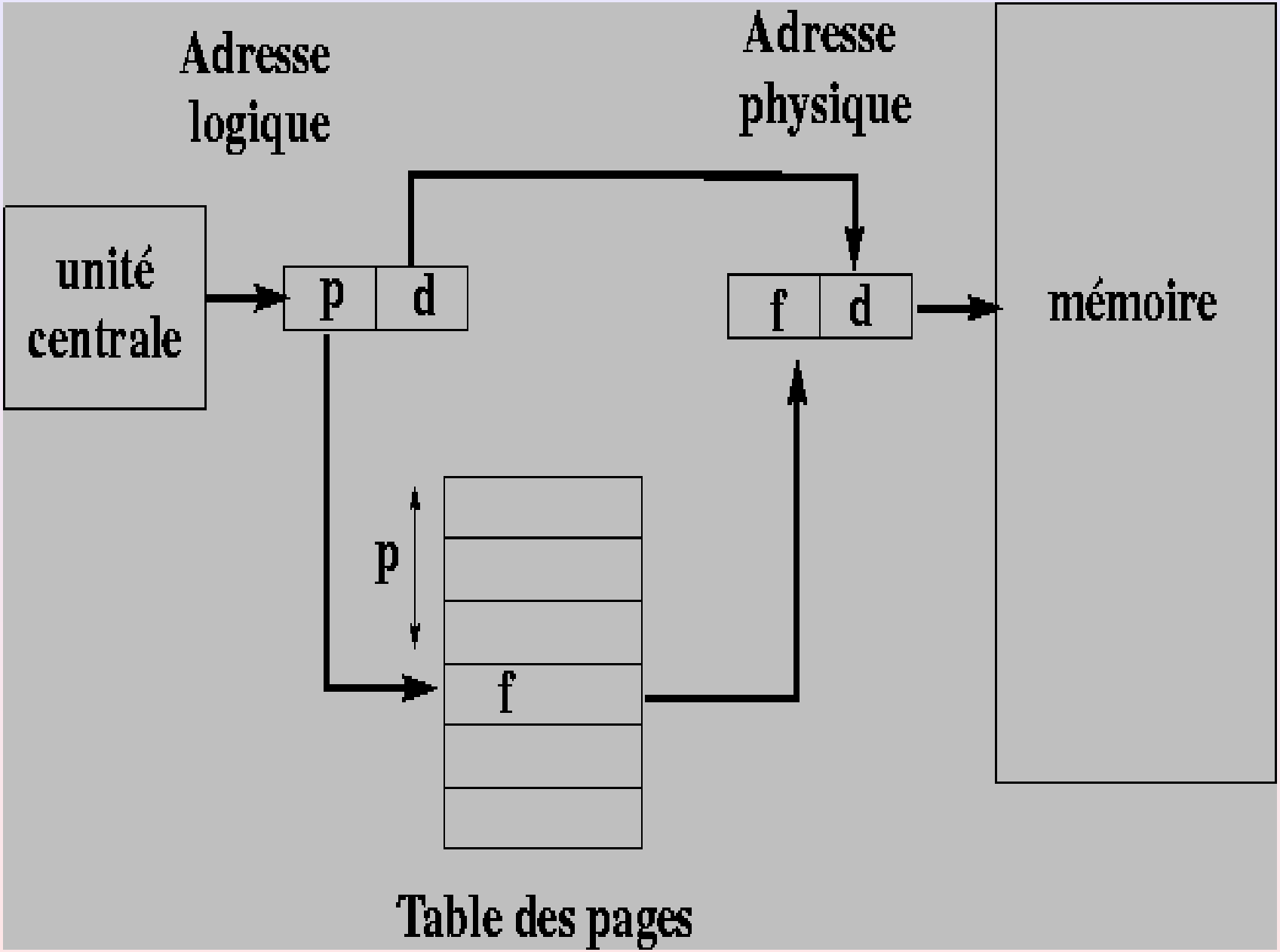
- La mémoire physique est découpée en cadres de page (même taille d'une page).
- Adresses physique est un couple (f, d), cadre de pages et déplacement.
- A chaque page logique p peut correspondre un cadre de page $f = f(p)$.
- Cette correspondance est maintenue dans la table des pages.
- La MMU (memory management unit) calcule un adresse physique à partir d'un adresse logique selon la formule

$$phys(p, d) = (f(p), d)\,.$$

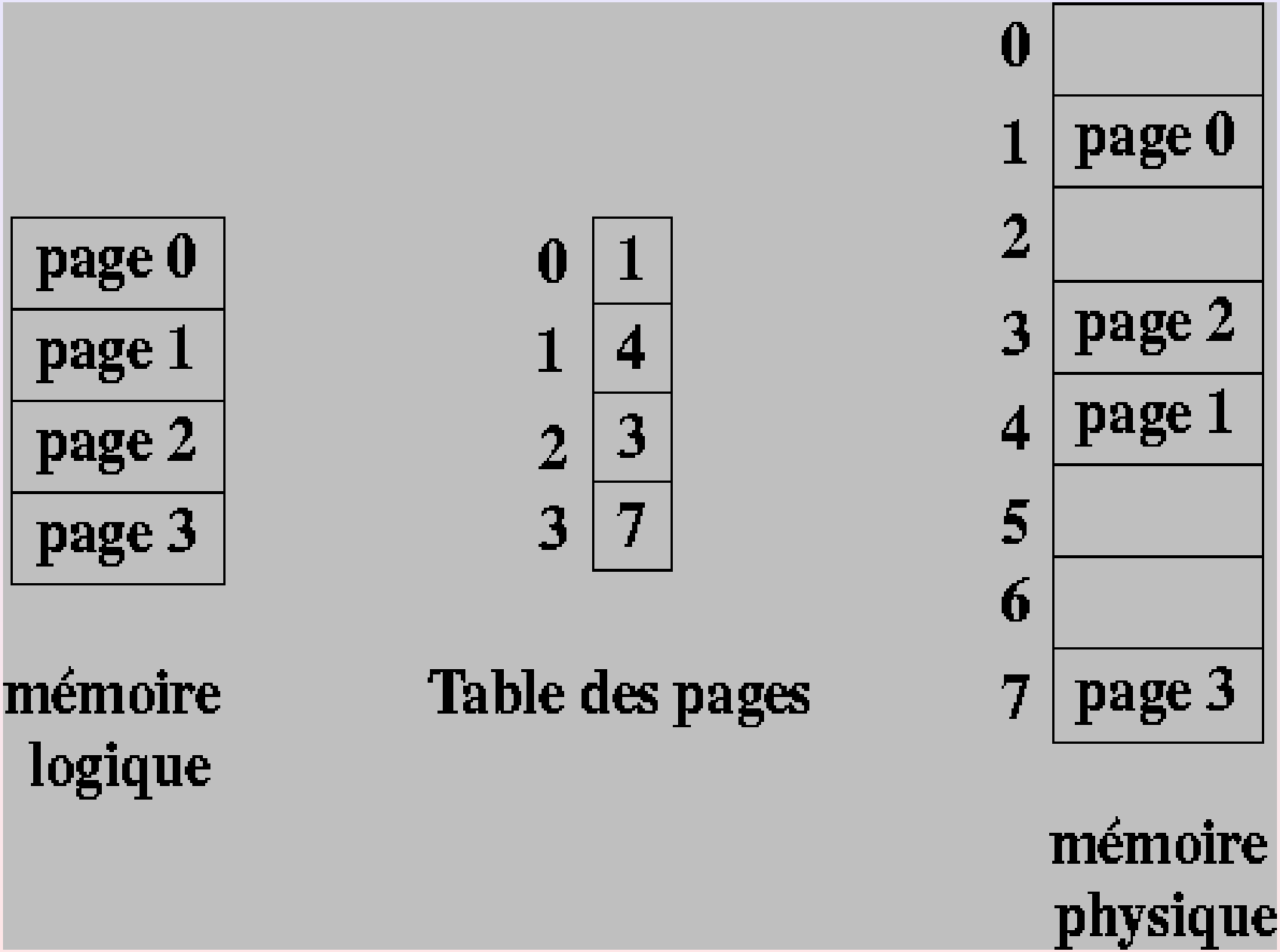
Gestion de la multiprogrammation :

- Le PTBR, registre de base de la table de pages, « page-table base register ».

Traduction des adresses logiques en adresses physiques



Exemple


page 0
page 1
page 2
page 3
mémoire logique
0 1
1 4
2 3
3 7
Table des pages
0
1 page 0
2
3 page 2
4 page 1
5
6
7 page 3
mémoire physique

Les tables de pages : temps d'accès

Problème :

- Le temps d'acces en mémoire est doublé.

Solution :

- Les TLBs, registres associatifs,
  « translation look-aside buffers »:
  - on y cache la correspondance $p \longrightarrow f(p)$,
  - recherche de la valeur $f(p)$ assez efficace,
  - haut coût de ce matériel.

Les tables de pages : temps d'accès

Problème :

- Le temps d'acces en mémoire est doublé.

Solution :

- Les TLBs, registres associatifs,
  « translation look-aside buffers »:
  - on y cache la correspondance $p \longrightarrow f(p)$,
  - recherche de la valeur $f(p)$ assez efficace,
  - haut coût de ce matériel.

Taux de présence et coût d'accès en mémoire

Taux de présence :

probabilité que la valeur $f(p)$ se trouve dans un TLB.

$$\text{taux de présence} = 0,80$$

$$\text{temps d'un accès en mémoire} = 100 \text{ nanosecondes}$$

$$\text{temps d'accès aux TLBs} = 20 \text{ nanosecondes}$$

$$\text{temps effectif d'accès en mémoire} = 0,8 * 120 + 0,2 * 220 \text{ nanosecondes}$$

Taux de présence et coût d'accès en mémoire

Taux de présence :

probabilité que la valeur $f(p)$ se trouve dans un TLB.

$$\text{taux de présence} = 0,80$$

$$\text{temps d'un accès en mémoire} = 100 \text{ nanosecondes}$$

$$\text{temps d'accès aux TLBs} = 20 \text{ nanosecondes}$$

$$\text{temps effectif d'accès en mémoire} = 0,8 * 120 + 0,2 * 220 \text{ nanosecondes}$$

Les tables de pages : utilisation de la mémoire

Remarque : une table par processus.

Problème : si

$$\text{adresse logique} \in \{0, \ldots, 2^{32} - 1\}$$
$$\text{taille d'une page} = 4K = 2^{12},$$

alors

$$\text{taille de la table de pages} = 2^{20}.$$

Solutions :

- plusieurs niveaux d'indirection,
- tables de pages inversées.

Les tables de pages : utilisation de la mémoire

Remarque : une table par processus.

Problème : si

$$\text{adresse logique} \in \{0, \ldots, 2^{32} - 1\}$$
$$\text{taille d'une page} = 4K = 2^{12}\,,$$

alors

$$\text{taille de la table de pages} = 2^{20}\,.$$

Solutions :

- plusieurs niveaux d'indirection,
- tables de pages inversées.

Les tables de pages : utilisation de la mémoire

Remarque : une table par processus.

Problème : si

$$\text{adresse logique} \in \{0, \ldots, 2^{32} - 1\}$$
$$\text{taille d'une page} = 4K = 2^{12}\,,$$

alors

$$\text{taille de la table de pages} = 2^{20}\,.$$

Solutions :

- plusieurs niveaux d'indirection,
- tables de pages inversées.

Plan

La mémoire segmentée

But :

- Partage des ressources (le code) entre plusieurs programmes.
- Organisation de la mémoire en unités logiques :
  - code (TEXT),
  - données statiques initialisés (DATA),
  - données statiques non initialisés (BSS),
  - données dynamiques (TAS),
  - pile d'exécution.

Solution à la fragmentation : on couple segmentation et pagination.

La mémoire segmentée

But :

- Partage des ressources (le code) entre plusieurs programmes.
- Organisation de la mémoire en unités logiques :
  - code (TEXT),
  - données statiques initialisés (DATA),
  - données statiques non initialisés (BSS),
  - données dynamiques (TAS),
  - pile d'exécution.

Solution à la fragmentation : on couple segmentation et pagination.

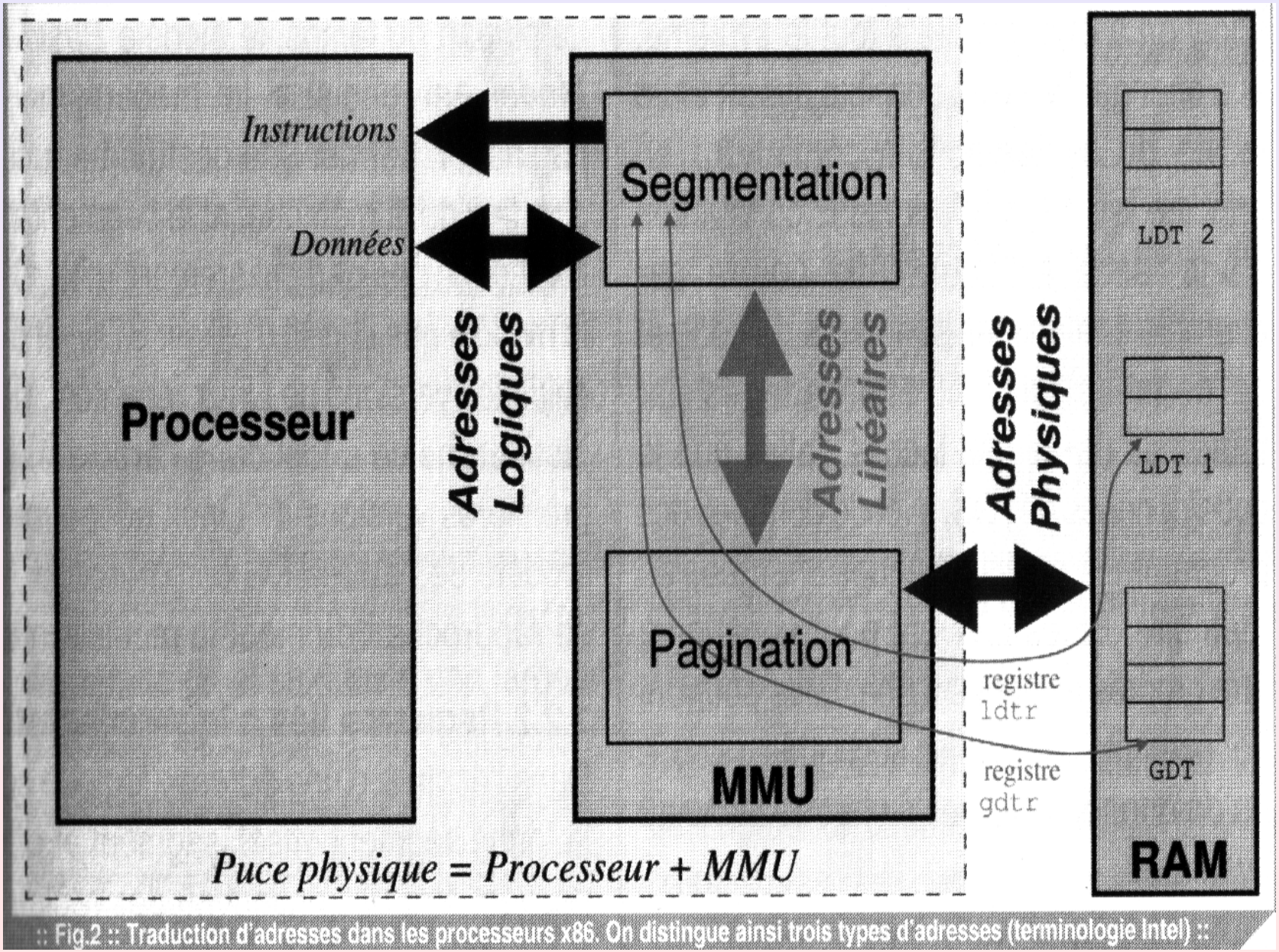
La mémoire segmentée

But :

- Partage des ressources (le code) entre plusieurs programmes.
- Organisation de la mémoire en unités logiques :
  - code (TEXT),
  - données statiques initialisés (DATA),
  - données statiques non initialisés (BSS),
  - données dynamiques (TAS),
  - pile d'exécution.

Solution à la fragmentation : on couple segmentation et pagination.

Exemple : segmentation et pagination sur un x86 (Linux magasine)



:: Fig.2 :: Traduction d'adresses dans les processeurs x86. On distingue ainsi trois types d'adresses (terminologie Intel) ::

. . . sur un x86

GDT : Global descriptor table. Description des adresses et attributs des segments partagés.
Appartient aux noyau.

LDT : Local descriptor table. Description des adresses et attributs des segments appartenant à un processus donné.
Appartient au processus.

`gdtr`, `ldtr` :
registres pour repérer la GDT et la LDT courante.

. . . sur un x86

GDT : Global descriptor table. Description des adresses et attributs des segments partagés.
Appartient aux noyau.

LDT : Local descriptor table. Description des adresses et attributs des segments appartenant à un processus donné.
Appartient au processus.

`gdtr`, `ldtr` :
registres pour repérer la GDT et la LDT courante.

. . . sur un x86

GDT : Global descriptor table. Description des adresses et attributs des segments partagés.
Appartient aux noyau.

LDT : Local descriptor table. Description des adresses et attributs des segments appartenant à un processus donné.
Appartient au processus.

`gdtr`, `ldtr` : registres pour repérer la GDT et la LDT courante.

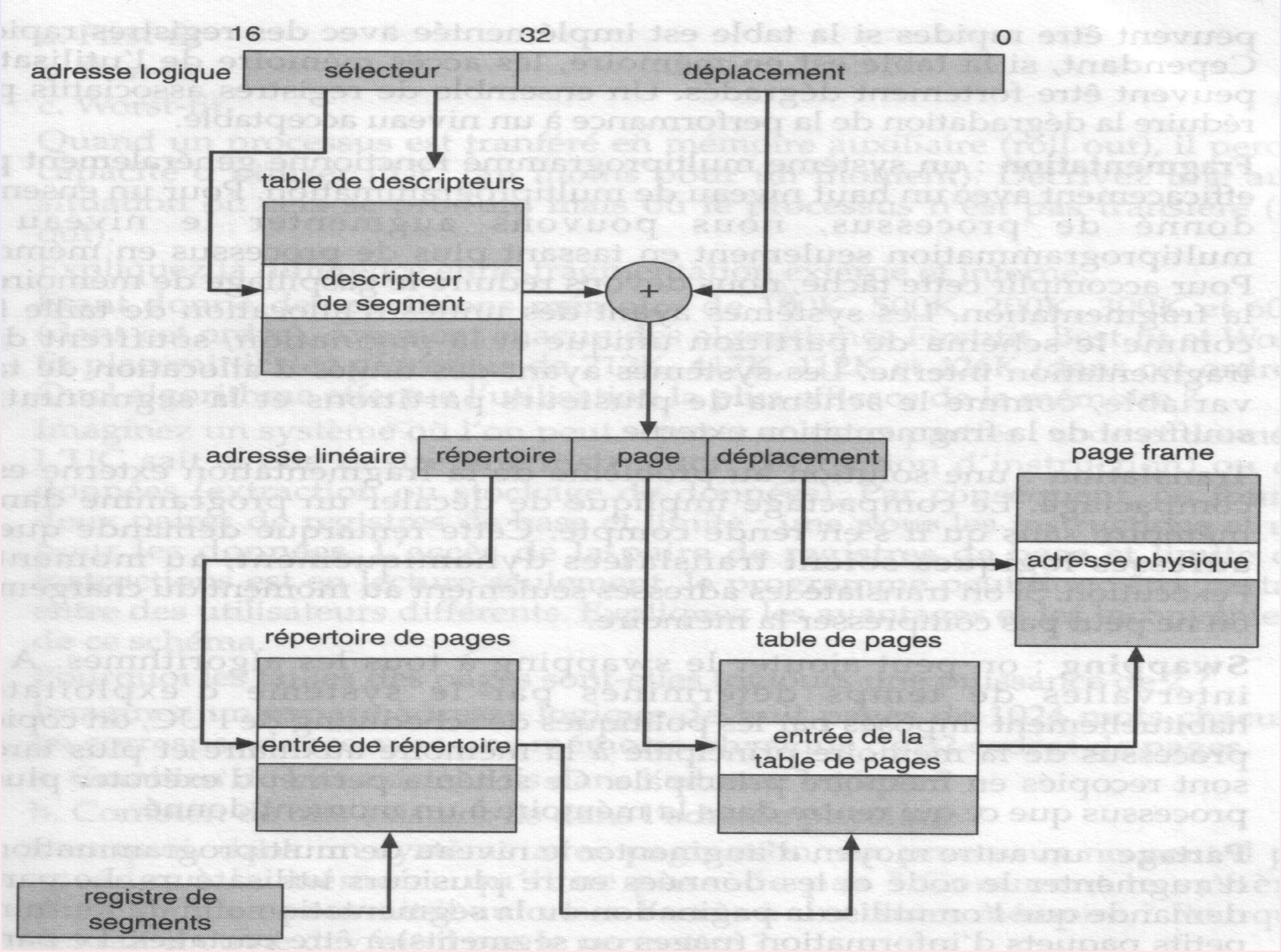
. . . sur un x86

GDT : Global descriptor table. Description des adresses et attributs des segments partagés.
Appartient aux noyau.

LDT : Local descriptor table. Description des adresses et attributs des segments appartenant à un processus donné.
Appartient au processus.

`gdtr`, `ldtr` : registres pour repérer la GDT et la LDT courante.

. . . sur un x86 (Silberschatz-Galvin)



Plan

Les limites du swap

Problèmes avec le swap :

- coût,
- pas possible avoir des processus plus grand que la mémoire vive.

Remarques :

- portions du code très souvent utilisés, portions du code peu souvent utilisés,
- de même, pour les données.

Les limites du swap

Problèmes avec le swap :

- coût,
- pas possible avoir des processus plus grand que la mémoire vive.

Remarques :

- portions du code très souvent utilisés, portions du code peu souvent utilisés,
- de même, pour les données.

Solutions

Les « overlays » :

- morceaux d'un programme
  chargé en mémoire de façon séquentielle.

Chargement dynamique :

- une fonction est chargée en mémoire seulement à son appel.

Problèmes:

- le travail est laissé au programmeur.

Solutions

Les « overlays » :

- morceaux d'un programme
  chargé en mémoire de façon séquentielle.

Chargement dynamique :

- une fonction est chargée en mémoire seulement à son appel.

Problèmes:

- le travail est laissé au programmeur.

Solutions

Les « overlays » :

- morceaux d'un programme
  chargé en mémoire de façon séquentielle.

Chargement dynamique :

- une fonction est chargée en mémoire seulement à son appel.

Problèmes:

- le travail est laissé au programmeur.

Plan

La pagination à la demande ou swappeur paresseux

- La mémoire logique (virtuelle) est découpée en pages.
- La mémoire virtuelle est potentiellement infinie.
- La mémoire physique est découpée en cadres de page.
- La mémoire physique est composée de :
  mémoire vive, mémoire sur support swap,
  mémoire sur partition disque.
- Un nombre restreint de pages est chargé en mémoire vive.
- Une entrée dans la table des pages peut être dans l'état:
  - valide : le cadre de pages est en mémoire vive,
  - invalide : le cadre de page se trouve sur disque.
    Pour y accéder il faut préalablement la charger en mémoire.

La pagination à la demande ou swappeur paresseux

- La mémoire logique (virtuelle) est découpée en pages.
- La mémoire virtuelle est potentiellement infinie.
- La mémoire physique est découpée en cadres de page.
- La mémoire physique est composée de :
  mémoire vive, mémoire sur support swap,
  mémoire sur partition disque.
- Un nombre restreint de pages est chargé en mémoire vive.
- Une entrée dans la table des pages peut être dans l'état:
  - valide : le cadre de pages est en mémoire vive,
  - invalide : le cadre de page se trouve sur disque.
    Pour y accéder il faut préalablement la charger en mémoire.

La pagination à la demande ou swappeur paresseux

- La mémoire logique (virtuelle) est découpée en pages.
- La mémoire virtuelle est potentiellement infinie.
- La mémoire physique est découpée en cadres de page.
- La mémoire physique est composée de :
  mémoire vive, mémoire sur support swap,
  mémoire sur partition disque.
- Un nombre restreint de pages est chargé en mémoire vive.
- Une entrée dans la table des pages peut être dans l'état:
  - valide : le cadre de pages est en mémoire vive,
  - invalide : le cadre de page se trouve sur disque.
    Pour y accéder il faut préalablement la charger en mémoire.

La pagination à la demande ou swappeur paresseux

- La mémoire logique (virtuelle) est découpée en pages.
- La mémoire virtuelle est potentiellement infinie.
- La mémoire physique est découpée en cadres de page.
- La mémoire physique est composée de :
  mémoire vive, mémoire sur support swap,
  mémoire sur partition disque.
- Un nombre restreint de pages est chargé en mémoire vive.
- Une entrée dans la table des pages peut être dans l'état:
  - valide : le cadre de pages est en mémoire vive,
  - invalide : le cadre de page se trouve sur disque.
    Pour y accéder il faut préalablement la charger en mémoire.

La pagination à la demande ou swappeur paresseux

- La mémoire logique (virtuelle) est découpée en pages.
- La mémoire virtuelle est potentiellement infinie.
- La mémoire physique est découpée en cadres de page.
- La mémoire physique est composée de :
  mémoire vive, mémoire sur support swap,
  mémoire sur partition disque.
- Un nombre restreint de pages est chargé en mémoire vive.
- Une entrée dans la table des pages peut être dans l'état:
  - valide : le cadre de pages est en mémoire vive,
  - invalide : le cadre de page se trouve sur disque.
    Pour y accéder il faut préalablement la charger en mémoire.

La pagination à la demande ou swappeur paresseux

- La mémoire logique (virtuelle) est découpée en pages.
- La mémoire virtuelle est potentiellement infinie.
- La mémoire physique est découpée en cadres de page.
- La mémoire physique est composée de :
  mémoire vive, mémoire sur support swap,
  mémoire sur partition disque.
- Un nombre restreint de pages est chargé en mémoire vive.
- Une entrée dans la table des pages peut être dans l'état:
  - valide : le cadre de pages est en mémoire vive,
  - invalide : le cadre de page se trouve sur disque.
    Pour y accéder il faut préalablement la charger en mémoire.

La pagination à la demande ou swappeur paresseux

- La mémoire logique (virtuelle) est découpée en pages.
- La mémoire virtuelle est potentiellement infinie.
- La mémoire physique est découpée en cadres de page.
- La mémoire physique est composée de :
  mémoire vive, mémoire sur support swap,
  mémoire sur partition disque.
- Un nombre restreint de pages est chargé en mémoire vive.
- Une entrée dans la table des pages peut être dans l'état:
  - valide : le cadre de pages est en mémoire vive,
  - invalide : le cadre de page se trouve sur disque.
    Pour y accéder il faut préalablement la charger en mémoire.

La pagination à la demande ou swappeur paresseux

- La mémoire logique (virtuelle) est découpée en pages.
- La mémoire virtuelle est potentiellement infinie.
- La mémoire physique est découpée en cadres de page.
- La mémoire physique est composée de :
  mémoire vive, mémoire sur support swap,
  mémoire sur partition disque.
- Un nombre restreint de pages est chargé en mémoire vive.
- Une entrée dans la table des pages peut être dans l'état:
  - valide : le cadre de pages est en mémoire vive,
  - invalide : le cadre de page se trouve sur disque.
    Pour y accéder il faut préalablement la charger en mémoire.

Accès à un adresses logique

On demande l'accès à un adresses logique :

- si la page se trouve en mémoire vive continuer,
- sinon, lever une interruption `Page Fault`, et
- traiter l'interruption : charger la page en mémoire vive,
- une fois que la page demandé est en place, executer à nouveaux l'opération qui a declanché le `Page Fault`.

Remarque :
une instruction doit être interruptible, par exemple :

```
add A B in C
```

avec un Page Fault sur l'accès à `C`.

Accès à un adresses logique

On demande l'accès à un adresses logique :

- si la page se trouve en mémoire vive continuer,
- sinon, lever une interruption `Page Fault`, et
- traiter l'interruption : charger la page en mémoire vive,
- une fois que la page demandé est en place, executer à nouveaux l'opération qui a declanché le `Page Fault`.

Remarque :
une instruction doit être interruptible, par exemple :

```
add A B in C
```

avec un Page Fault sur l'accès à `C`.

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
  restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
  restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
  restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
- restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
- restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
- restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
- restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
- restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
- restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur, restauration du contexte du processus

Traitement du Page Fault

Erreur de page provoque :

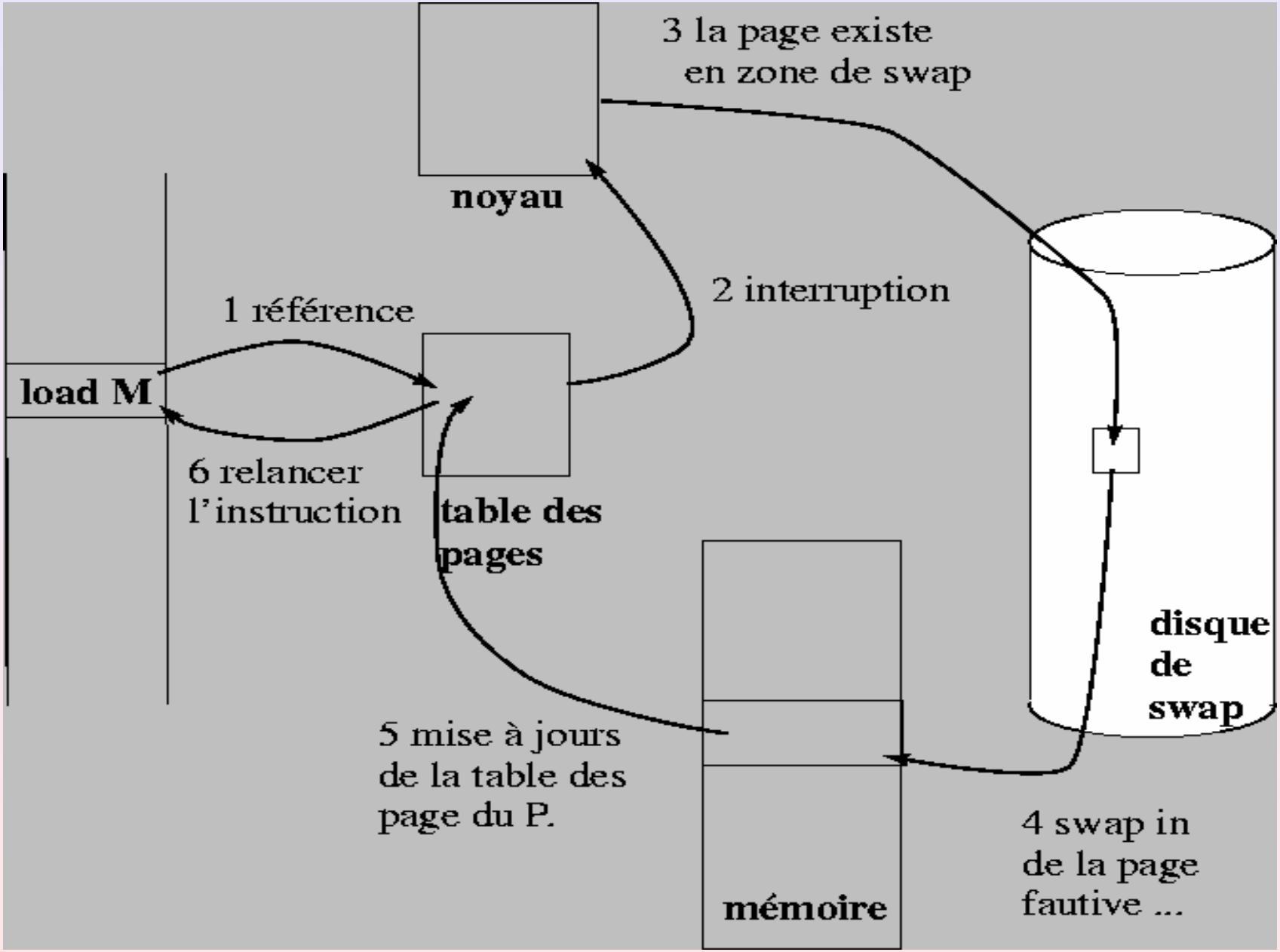
- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur, restauration du contexte du processus.

Traitement du Page Fault

Erreur de page provoque :

- interruption,
- sauvegarde du contexte,
- reconnaissance erreur de page,
- déterminer où la page se trouve sur la mémoire secondaire,
- charger la page dans en endroit libre, ou remplacer une page,
- attente du transfert de la périphérique, et allocation la CPU à quelques autre processus,
- interruption périphérique,
- sauvegarde du contexte du processus,
- reconnaissance interruption périphérique,
- mise à jour table des pages,
- état : en attente de se dérouler,
- quand choisi par l'ordonnanceur,
  restauration du contexte du processus.

Traitement d'un Page Fault (Revuz 1998)



Coût du Demand Paging

Calcul du coût :

$$\text{temps effectif} = (1 - p) * ma + p * (\text{temps traitement})$$

où

$$p = \text{probabilité du Page Fault}$$
$$ma = \text{temps acces à la mémoire}$$

Exemple : si

$$ma = 1 \text{ microseconde}$$
$$\text{temps traitement} = 5000 \text{ microsecondes}$$
$$p = 1/1000 \quad \text{(un Page Fault chaque 1000 accès)}$$

alors

$$\text{temps effectif} = (1 - p) + p * 5000 = 5,999$$

Coût du Demand Paging

Calcul du coût :

$$\text{temps effectif} = (1 - p) * ma + p * (\text{temps traitement})$$

où

$$p = \text{probabilité du Page Fault}$$
$$ma = \text{temps acces à la mémoire}$$

Exemple : si

$$ma = 1 \text{ microseconde}$$
$$\text{temps traitement} = 5000 \text{ microsecondes}$$
$$p = 1/1000 \quad \text{(un Page Fault chaque 1000 accès)}$$

alors

$$\text{temps effectif} = (1 - p) + p * 5000 = 5,999$$

Coût du Demand Paging

Calcul du coût :

$$\text{temps effectif} = (1 - p) * ma + p * (\text{temps traitement})$$

où

$$p = \text{probabilité du Page Fault}$$
$$ma = \text{temps acces à la mémoire}$$

Exemple : si

$$ma = 1 \text{ microseconde}$$
$$\text{temps traitement} = 5000 \text{ microsecondes}$$
$$p = 1/1000 \quad \text{(un Page Fault chaque 1000 accès)}$$

alors

$$\text{temps effectif} = (1 - p) + p * 5000 = 5,999$$

Plan

Contexte

- Un accès en mémoire déclenche un `page fault`.
- Il faut allouer un cadre de page libre en mémoire vive.
- Tous les cadres de page sont occupés.
- *Choisir une victime* :
  un cadre à déplacer en mémoire secondaire.
- On libère ce cadre, on y transfère le cadre demandé.

Remarques :

- Le nombre de remplacements augmente avec le niveaux de multiprogrammation.
- Un remplacement nécessite deux transferts vers/du disque :
  si le cadre n'est pas *sale* et s'il existe une copie sur disque,
  alors il n'est pas nécessaire le recopier sur disque.

Contexte

- Un accès en mémoire déclenche un `page fault`.
- Il faut allouer un cadre de page libre en mémoire vive.
- Tous les cadres de page sont occupés.
- *Choisir une victime* :
  un cadre à déplacer en mémoire secondaire.
- On libère ce cadre, on y transfère le cadre demandé.

Remarques :

- Le nombre de remplacements augmente avec le niveaux de multiprogrammation.
- Un remplacement nécessite deux transferts vers/du disque :
  si le cadre n'est pas *sale* et s'il existe une copie sur disque,
  alors il n'est pas nécessaire le recopier sur disque.

Objectifs

Minimiser le taux de remplacement.

Remplacement FIFO

La page la *plus ancienne* est remplacée.

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	231	5

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	231	5
0	230	6

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	231	5
0	230	6
3	230	6

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	231	5
0	230	6
3	230	6
0	230	6

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	231	5
0	230	6
3	230	6
0	230	6
4	430	7

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	231	5
0	230	6
3	230	6
0	230	6
4	430	7
2	420	8

Remplacement FIFO

La page la *plus ancienne* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	231	5
0	230	6
3	230	6
0	230	6
4	430	7
2	420	8
1	421	9

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.

État des pages en mémoire au début : xxx

Page démandée Pages en mémoire (après) No Page Faults

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5
4	243	6

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5
4	243	6
2	243	6

Remplacement OPTIMAL

La page *referencée plus tard* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5
4	243	6
2	243	6
1	241	7

Remplacement LRU

La page *moins recemment utilisée* est remplacée.

État des pages en mémoire au début : xxx

Page démandée Pages en mémoire (après) No Page Faults

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5
4	403	6

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5
4	403	6
2	402	7

Remplacement LRU

La page *moins recemment utilisée* est remplacée.
État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5
4	403	6
2	402	7
1	412	8

Remplacement Deuxième Chance

- Chaque cadre de page possède un *bit de référence* (niveaux matériel).
- Si un cadre est referencé, son bit est placé à 1.
- Un algorithme de type FIFO est utilisé.
- La tète de la file est remplacée, si son bit est à 0.
- Si le bit de la tète est 1, en met ce bit à 0, et on place ce cadre en queue.

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5
4	403	6

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5
4	403	6
2	423	7

Exemple deuxième chance

État des pages en mémoire au début : xxx

Page démandée	Pages en mémoire (après)	No Page Faults
7	7xx	1
0	70x	2
1	701	3
2	201	4
0	201	4
3	203	5
0	203	5
3	203	5
0	203	5
4	403	6
2	423	7
1	421	8