Chapitre 6 : Mémoire Virtuelle

Section 6.1 : La mémoire virtuelle

Une adresse encodée sur n bits peut avoir maximum 2^n

- 32 bits -> 4 Go de RAM
- 64 bits -> 18.446.744 To de RAM

Pour fonctionner correctement, 2 processus doivent accéder à des zones mémoires différentes. d'où l'invention de la mémoire virtuelle.
Mémoire virtuelle :

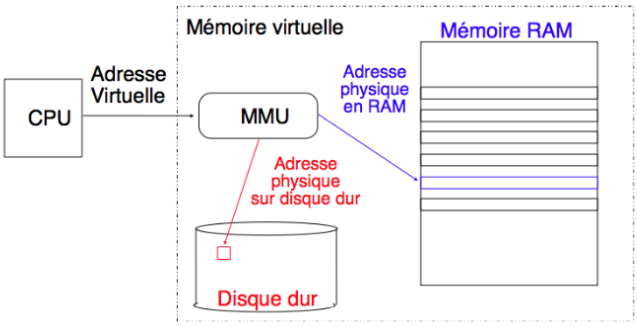
- **Adresse physique** : adresse utilisée par les puces pour des opérations de lecture et d'écriture
- **Adresse virtuelle** : adresse utilisée à l'interrieur d'un processus

Cette traduction est effectuée par le **MMU** (*Memory Management Unit*) intégré au CPU.

6.1.1 La mémoire virtuelle

Avantages

- Pas d'obligation d'avoir le même nombre de bits que les adresses plysiques
  - pc cheap avec moins de ram que ce que le système ne peut gérer
  - serveur avec plus de ram que ce que le système n'est capable de gérer
- partage de la mémoire entre les processus (distinction)
- peut alloier certaines parties de la mémoire pour qu'elle soit communes
- peut utiliser des dispositifs physisques comme si il s'agissait de dispositifs RAM

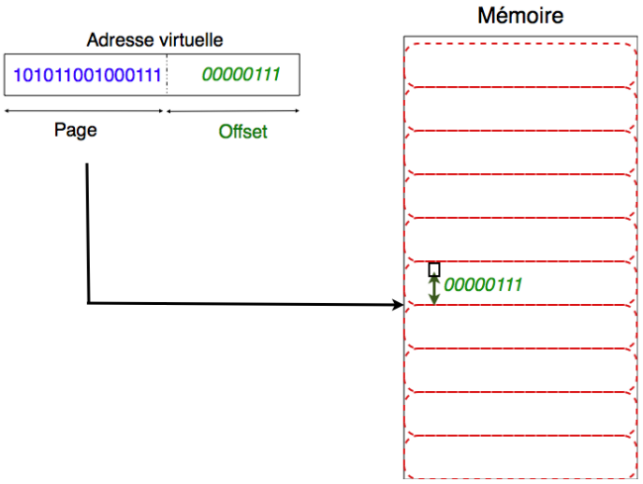


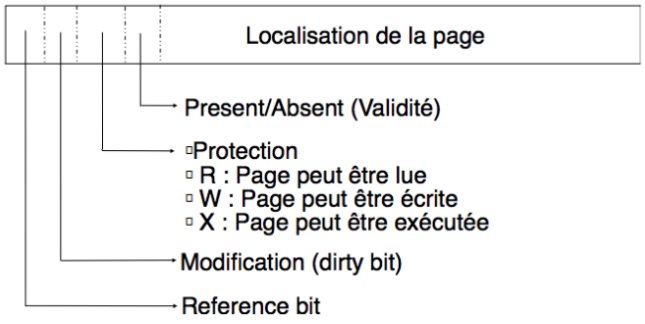
6.1.2 Fonctionnement de la mémoire

Coordination entre la ram et les dispositifs de stockages

- **RAM** : Lecture et écriture à n'importe quelle adresse
- **Dispositif de Stockage** : ensemble de secteurs, identifié par adresses
    - nécéssite au moins 512 Octets de lecture/écriture
        - moins rapide que la RAM

La mémoire virtuelle utilise des unités intermédiaires pour les adresses : Les *pages* une **page** : zone de mémoire contigue (sa taille dépend de l'architecture, de L'OS mais généralement c'est 4096 Octets)





Pour faire la traduction des pages en adresses : 1. Table des pages en RAM : * une ligne par page * bit de validité (page présente ou non en ram) * Adresses physiques 2. En C, cette table est stockée sous la forme d'un tableau. celui-ci est

capable de gérer n'importe quelle taille d'adresses

Lors d'un changement de contexte, l'OS rechargera la page contenant les informations utiles dans ses répertoires.

Performances

L'utilisation de mémoire virtuelle à un impact très négatif sur les performance. mais ce problème est lissé via l'utilisation de buffer à table de pages.

Contôle de la table des pages

Le contrôle de la table des pages est la responsabilité exclusive de l'OS.

6.1.3 Utilisation des dispositifs de stockage

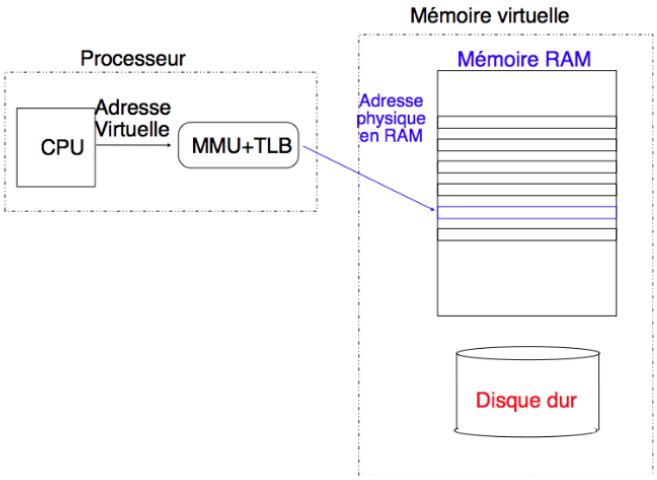
En Pratique , la RAM peut jouer le rôle de cache pour la mémoire virtuelle (les pages fréqemment utilisées sont sauvegardées en RAM et les peu fréquentes sur le disque).

Accès à la mémoire

- bit de validité vrai
  - permissions ok : accès direct
  - permissions fausses : interruption
- bit de validité faux
  - permissions ok : charge en RAM
  - permissions fausses : interruption et segmentation fault

Swap

Il s'agit de la partition d'un disque utilisé pour stôcker les pages comme de la mémoire virtuelle. La swap peut être crée avec mkswap, activée avec swapon. mais généralement, elle est lancée automatiquement au démarrage du système.



On peut aussi créer des fichiers swap. pour retenir la localisaition, ceux ci se composent de :

- dispositif de stockage
- inode
- offset

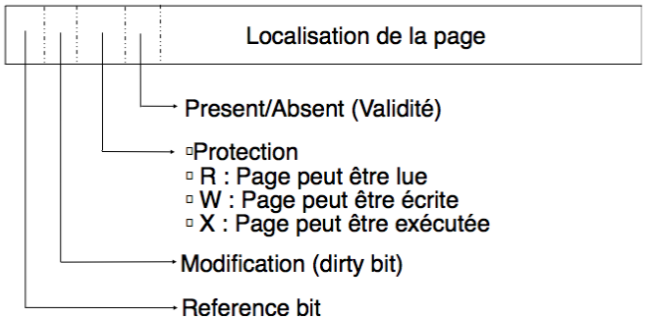
ils sont moins rapide que la swap car pas forcément contigu

6.1.4 Stratégie du remplacement des pages

Pour éviter l'encombrement de la ram, l'algorithme de remplacement peut décider de faire migrer des pages de la ram vers la swap. Tant que toute la RAM n'est pas remplie on préferera y stocker les données (localité de l'info)

Une fois que toute la mémoire est pleine, il faut désencombrer la RAM il existe 2 techniques:

1. à l'aide d'une liste fifo. on retire l'élément accédé en dernier
    - facile à implémenter
    - pas plus optimisé
2. sauvegarder le nombre d'accès à des pages et les charger en fonction de leur réccurence
3. utiliser des bits supplémentaires dans les pages



- **Bit de Référence** : mis à vrai par le MMU
- **Bit de modification** : mis à vrai à chaque écriture qui est relié à cette page

Ces deux bits sont mis à jour par le MMU dans le TLB. Ce dernier effectue une espèce de write-back

6.1.6 Fichiers mappés en mémoire

nmap : appel système permettant de rendre des pages d'un fichier accessible à travers la table des pages du processus.

> Rend accessible une portion d'un fichier via la mémoire (skip)