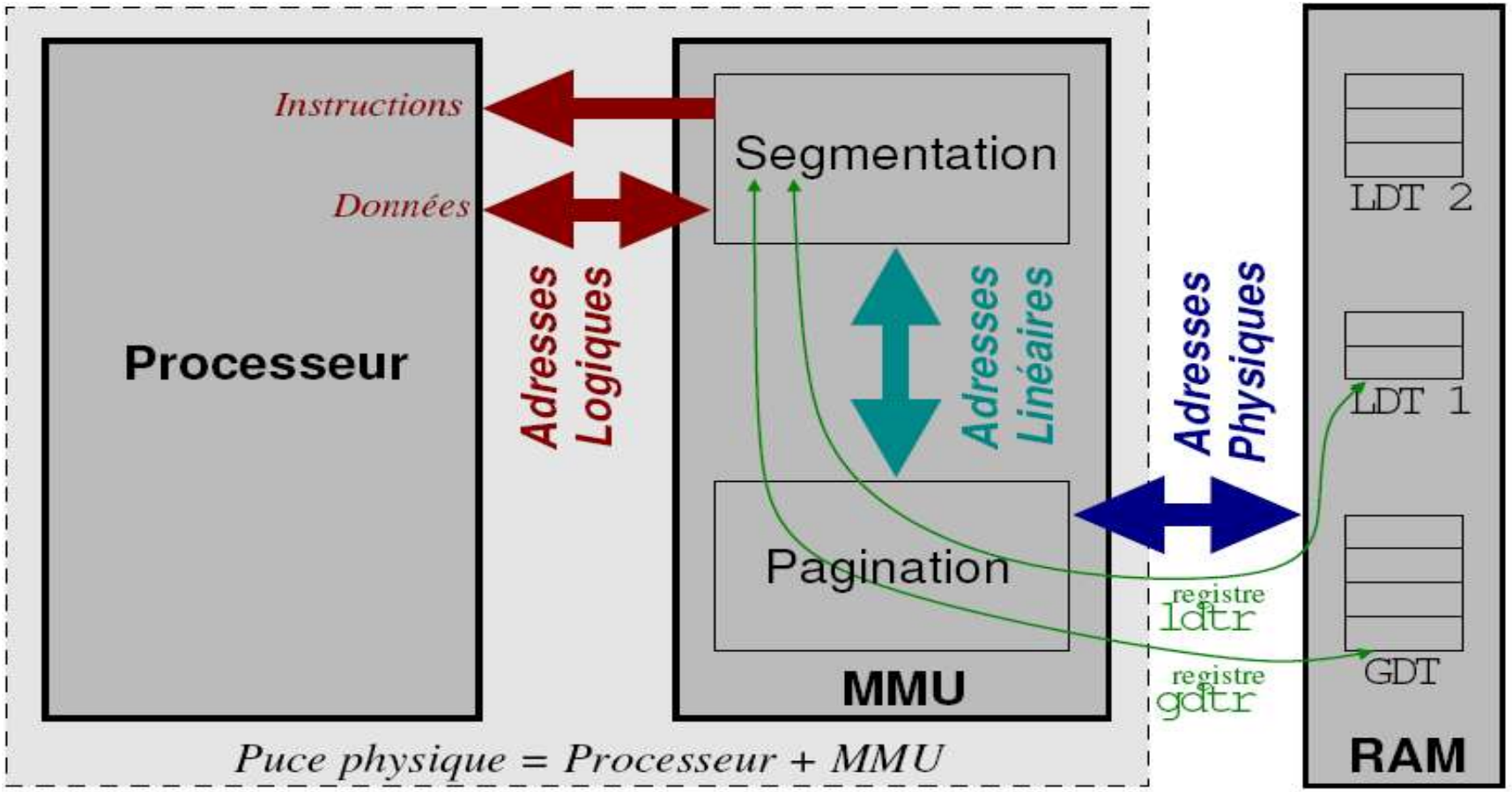
SOS
-
Gestion de la mémoire physique

Rappel



Traduction d'adresses dans les processeurs x86

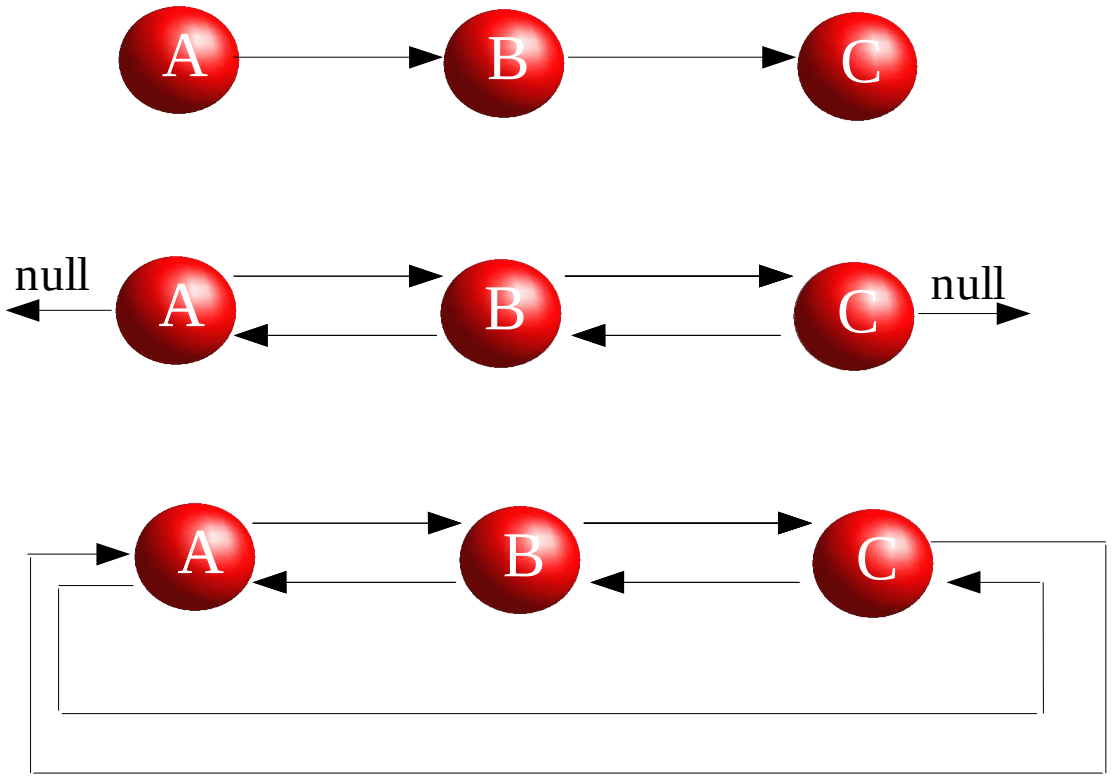
Sommaire

Introduction

- Mécanisme d'allocation de la mémoire pour le noyau :
  - Gestion de la mémoire physique
  - Pagination
  - Allocateur d'objets de taille arbitraire
- Principe assez simple
  - Gestion des emplacements vides
  - Gestion des emplacements utilisés
  - Pas de réallocation sans libération

1. Gestion des listes

- 1.1 Rappels sur les listes chainées



1.2 Macros, exemples

- Utilisation des macros :
  - gestion générique pour tout type de donnée
  - vérification du typage par le compilateur

- Exemple :
  - list_init(list) : initialise une liste vide
  - list_pop_head(list) : retourne et retire l'élément en tête
  - list_foreach_forward(list, iterator, nb_elements) : parcours des éléments de la tête jusqu'à la queue
  - list_collapse(list, iterator) : parcours et retire les éléments

2. Gestionnaire de mémoire physique

2.1 Macros fondamentales

- #define SOS_PAGE_SIZE (4*1024)
- #define SOS_PAGE_SHIFT 12 // 4 kB = 2 ^12
- #define SOS_PAGE_MASK ((1<<12) - 1)

- #define SOS_PAGE_ALIGN_INF(val)
- #define SOS_PAGE_ALIGN_SUP(val)

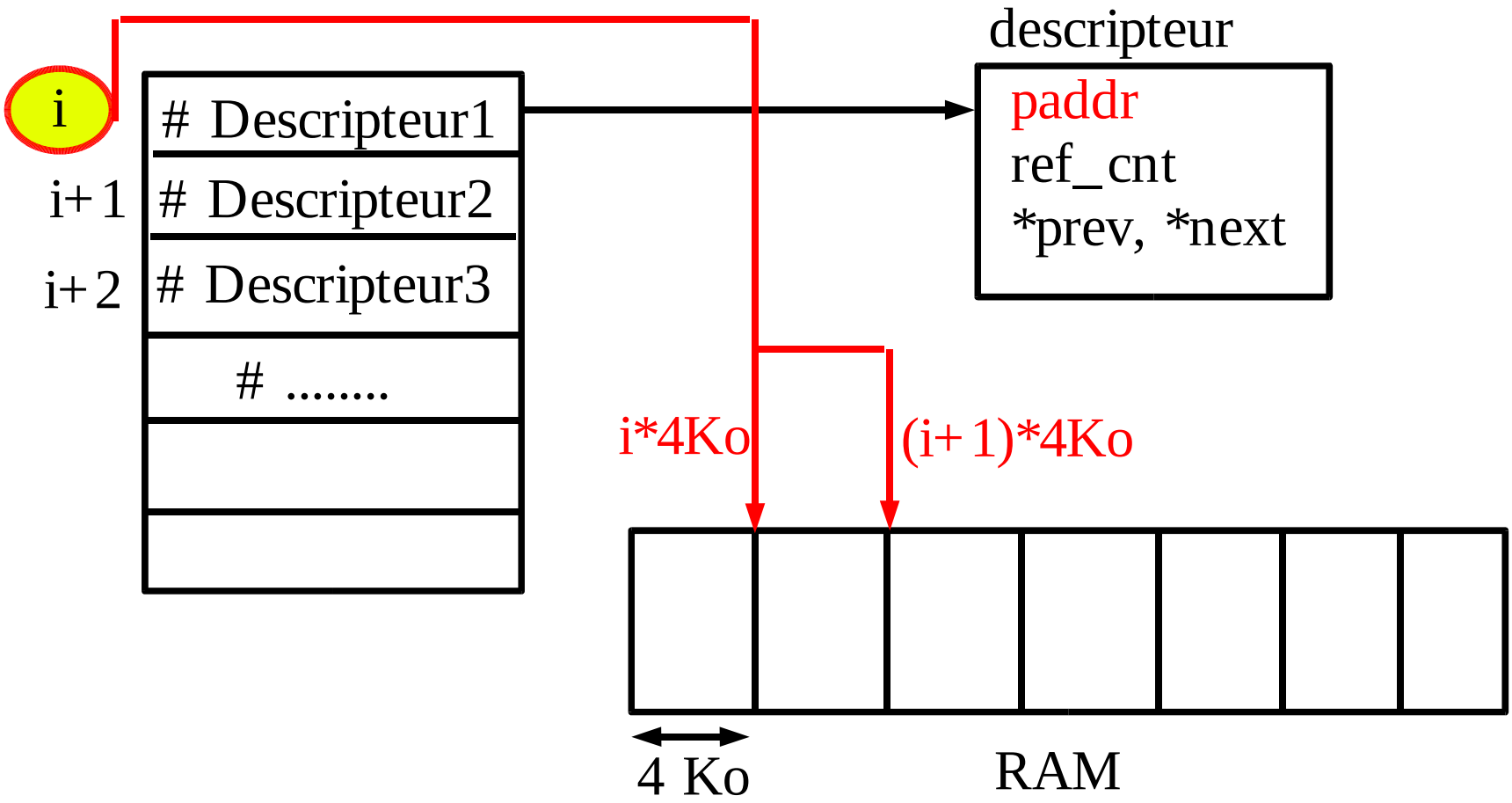
2.2 Descripteurs de pages physiques

- Pour chaque page physique : un descripteur de page physique

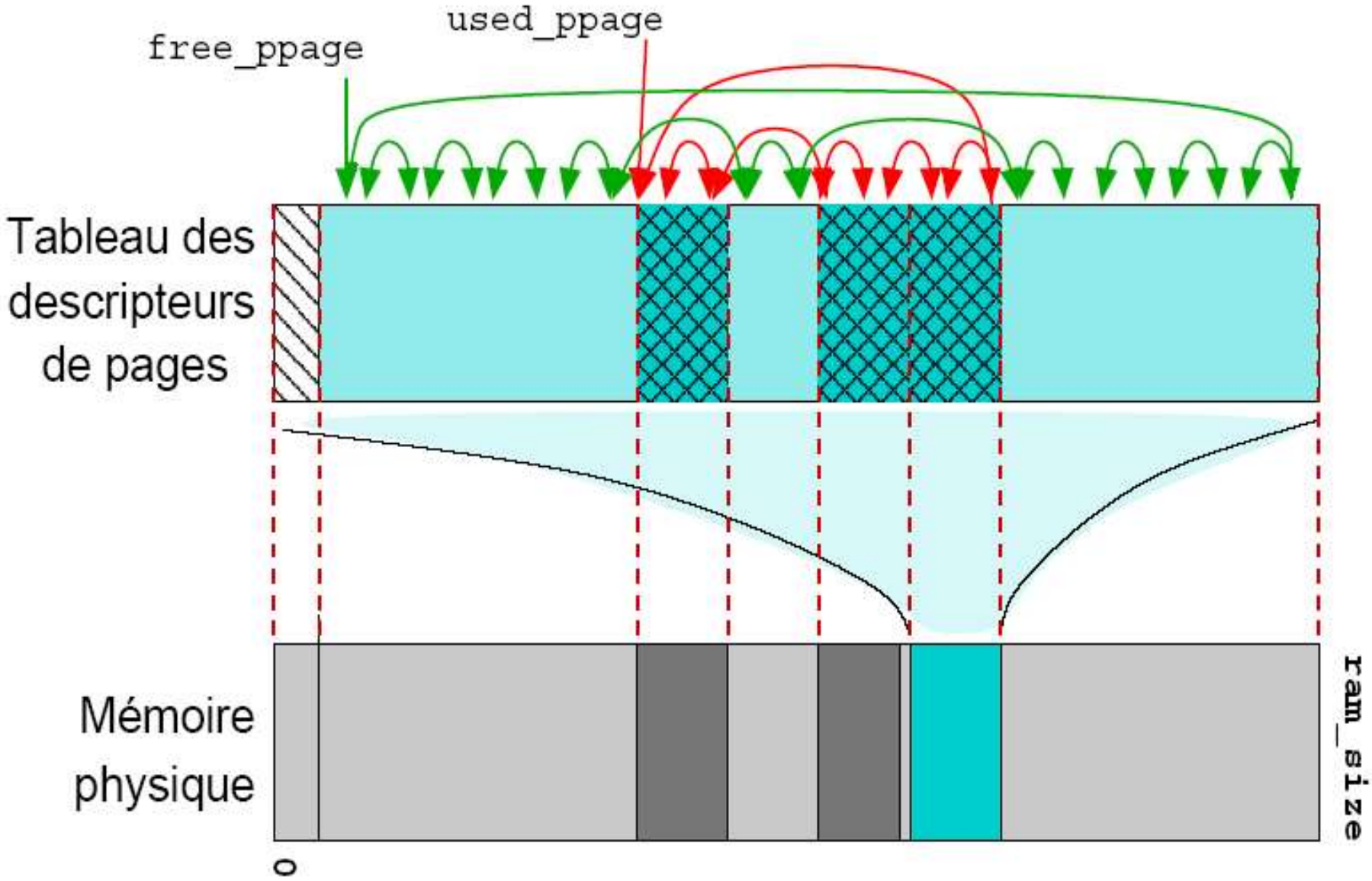
- Structure du descripteur :

```
struct physical_page_descr{
    sos_paddr_t paddr;
    sos_count_t ref_cnt;
    struct physical_page_descr *prev, *next;
};
```

- Tableau des descripteurs de pages physiques :
  - Taille = nb de pages physiques * taille d'un descripteur
  - Descripteur i du tableau est associé à la page physique [indice*4, (indice+1)*4]



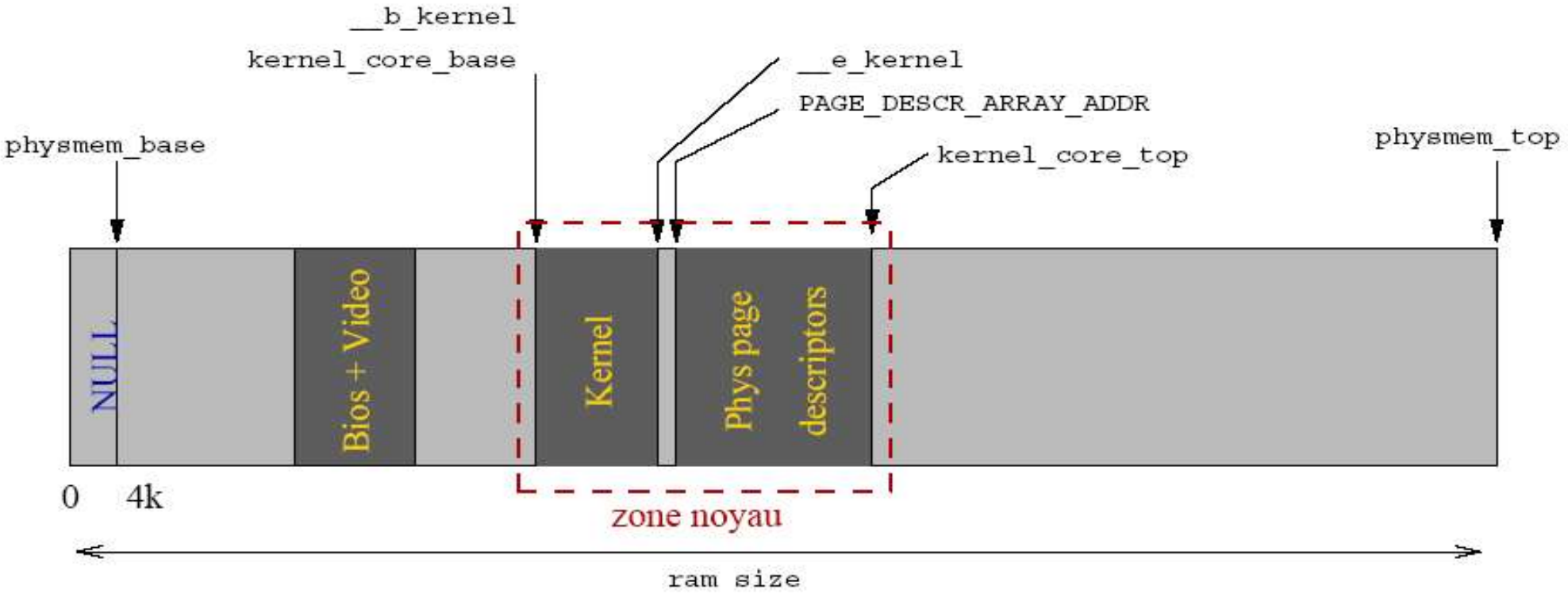
- Tableau et listes des descripteurs de pages physique



2.3 Initialisation

- Fonction sos_physmem_setup() :
  - prend la taille de la mémoire physique disponible
  - retourne les adresses physique début et fin de la zone noyau (resp kernel_core_base et kernel_core_top).
- La zone noyau contient :
  - Exécutable du noyau chargé par Grub
  - Tableau des descripteurs des pages physiques
- Tableau des descripteurs de pages physiques :
  - Se trouve juste après l'exécutable du noyau
  - Adresse de fin de la zone noyau = adresse de fin de l'exécutable + taille du tableau

- Début et fin de la mémoire physique :
  - physmem_base = adresse de début mémoire physique
  - physmem_top = adresse de fin mémoire physique
  - adresse de début ne commence pas à 0 (0 signifiera que la page est indisponible).
- Carte de la mémoire physique :



- Et on peut commencer l'initialisation :

  - Pour chaque descripteur, l'adresse physique de la page correspondante est calculée

  - Pages libres :
    - initialisation du compteur de référence à 0
    - ajout dans la liste des pages libres

  - Pages Occupées :
    - initialise le compteur de référence à 1
    - ajout dans la liste des pages occupées

  - Pages réservées :
    - seule la première page est réservée
    - aucune action

2.4 Allocation

- Allocation page par page

- Étapes :
  - Récupère la première page libre dans la liste des pages libres
  - Suppression de la liste libre
  - Incrémente son compteur de référence
  - Ajoute son descripteur dans la liste pages occupées
  - Retourne l'adresse de la page allouée

2.5 Déréférencement d'une page

- Décrémente le compteur de référence
- Si compteur de référence = 0
  - Suppression de la liste des pages occupées
  - Ajout dans la liste des pages vides
- Renvoi :
  - TRUE la page est libéré
  - FALSE toujours utilisée
  - Sinon code d'erreur négatif

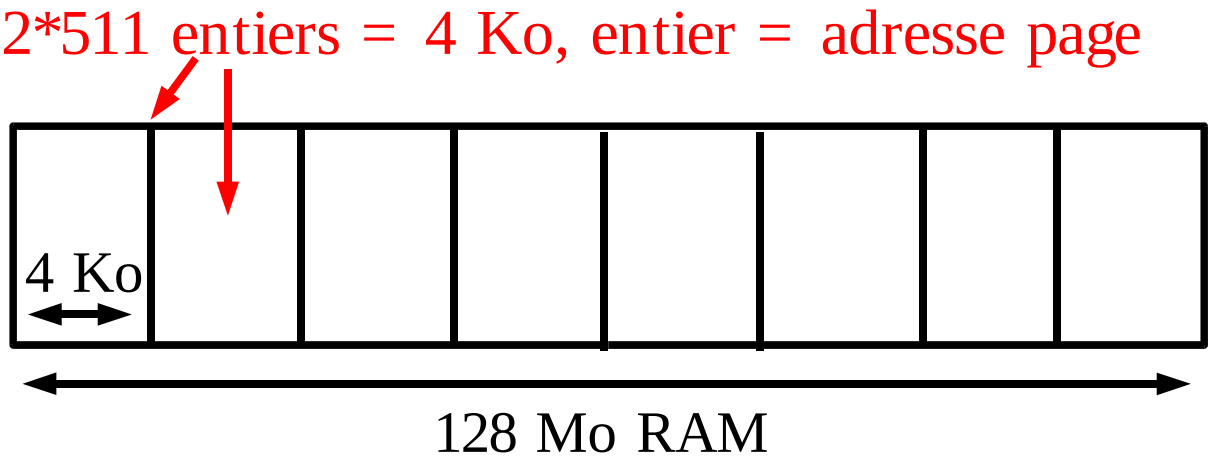
```
sos_ret_t sos_physmem_unref_physpage(sos_paddr_t ppage_paddr) {

  sos_ret_t retval = FALSE;

  struct physical_page_descr *ppage_descr = get_page_descr_at_paddr(ppage_paddr);

  if (! ppage_descr)
    return -SOS_EINVAL;

  if (ppage_descr->ref_cnt <= 0)
    return -SOS_EINVAL;

  ppage_descr->ref_cnt--;
  if (ppage_descr->ref_cnt <= 0){
      list_delete(used_ppage, ppage_descr);
      physmem_used_pages --;
      list_add_head(free_ppage, ppage_descr);

      retval = TRUE;
    }

  return retval;
}
```

2.6 Précautions d'usage

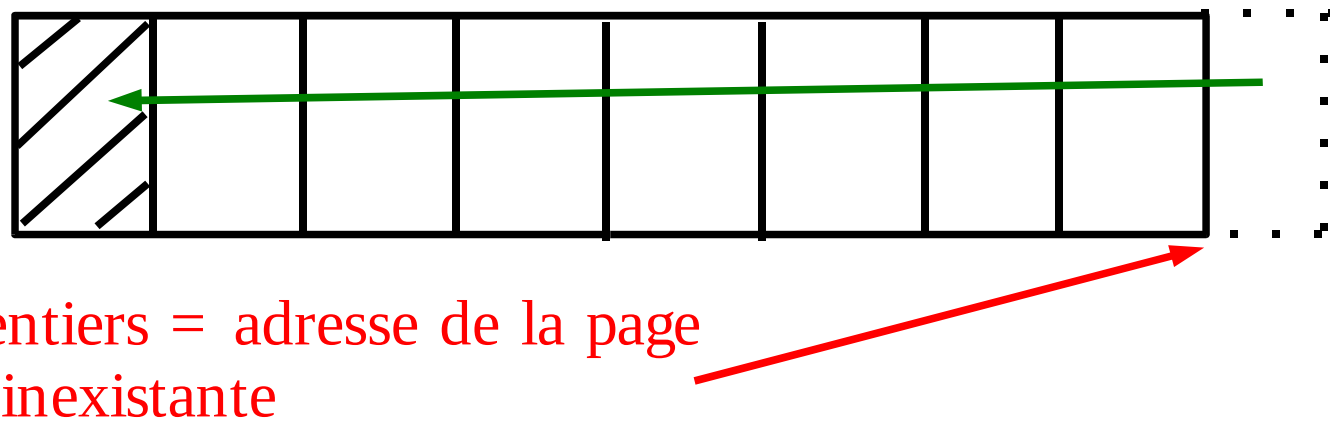
- Ces fonctions permettent d'allouer et de libérer de la mémoire
  - Mêmes précautions que pour malloc / free
- Gestion logique des ressources :
  - Les données sont toujours physiquement présentes
  - Accès à des adresses physiques non utilisées :
    - Noyau ne plantera pas immédiatement
    - Écrasement des données tôt ou tard
- Pagination :
  - Permet de détecter au plus tôt une partie de ces bogues latents.

2. Testons !

Test OK



Test KO



FIN

enfin. . . place à la pagination ;)