Segmentation

Attention : Dans ce TD, toutes les *tailles*, *adresses* et *numéros de segments* sont donnés en hexadécimal.

Exercice 1. Questions de cours Les questions de cours sont à destinées à vous permettre de vérifier votre compréhension du cours. Elles sont à travailler à l'avance et ne seront pas traitées en TD ou TP.

1. Expliquez ce qu'est l'édition de liens (en quelques mots).
2. Quelle est la meilleure stratégie d'allocation dans le cas d'une allocation contigüe ?
3. Donnez deux avantages de la segmentation.
4. Comment est déterminée une erreur de segmentation ?
5. En quoi consiste la pagination de la mémoire ?
6. Quel est le principal problème de la pagination ?
7. Quel est l'interet de la pagination hierarchique ?
8. Donnez deux avantages de la mémoire virtuelle.

Exercice 2. Mémoire contiguë On considère un système muni de 64ko de mémoire et capable de gérer plusieurs processus en mémoire simultanement.

1. Quelle est la taille de l'adresse physique ?
2. Quelle est la taille de l'adresse logique ?

On décide d'utiliser un algorithme d'allocation contiguë qui met en attente les processus lorsqu'il n'y a plus de bloc libre de taille suffisante. On considère l'exécution suivante, pour laquelle nous donnons pour chaque processus sa date d'arrivée, son besoin en mémoire (donné à la création du processus) et sa durée d'exécution totale :

Processus	Arrivée	Mémoire	Durée
P1	0	16ko	10
P2	2	8ko	4
P3	2	32ko	12
P4	4	12ko	12
P5	8	4ko	6
P6	12	32ko	4

3. On suppose que l'allocation se fait suivant un algorithme de type "First Fit". Décrivez précisément l'allocation mémoire à chaque pas de temps, en expliquant au fur et à mesure comment sont positionnés les processus.
4. Quel est le temps total d'exécution ?
5. Le taux de fragmentation est la quantité de mémoire dans les "trous" de mémoire libre ramenée à la quantité de mémoire actuellement allouée. Quel est le taux de fragmentation au temps t_8, t_{10} et t_{12} ?
6. L'exécution serait-elle meilleure avec un algorithme de type "Best Fit" ou "Worst Fit" ?
7. Quelle serait l'allocation optimale afin de minimiser le temps de calcul sur cet exemple ?

Exercice 3. Segmentation simple On considère un système de gestion de la mémoire utilisant la segmentation avec des adresses logiques sur 16bits dont les 4bits de poids fort indiquent le numéro de segment. Les premières lignes de la table des segments d'un processus sont données dans le tableau suivant :

Segment	Base	Limite
0	02B9	5FF
1	2A00	014
2	0090	100
3	C3A7	5D0
4	1D52	09F

1. Quelle quantité de mémoire un processus peut-il utiliser au maximum ?

2. Quelle est la taille maximale d'un segment ?

3. Quelles sont les adresses physiques correspondant aux adresses logiques suivantes : `0x0430`, `0x1010`, `0x2B0F`, `0x34FF` et `0x4100` ?

4. En supposant que toutes les autres lignes de la table ont les valeurs base et limit à zéro, quelle quantité de mémoire est allouée à ce processus.

Exercice 4. Mini segmentation On considère un système de gestion de la mémoire pour un micro-controleur, c'est-à-dire un très petit processeur consommant très peu d'énergie dont le prix est très bas. Ce système étant très simple, ses capacitées sont limitées. Il dispose quand même d'une gestion de la mémoire segmenté mais un peu particulière.

Ce système dispose de 16ko de mémoire physique. Afin de réduire la consomation mémoire, une seule table des segments est utilisée pour l'ensemble des processus. Cette table permet de décrire 128 segments, chaque ligne est composé de :

- 3bits indiquant le numéro du processus à qui appartient le segment décris (le numéro 7 indiquant un segment partagé par tous les processus) ;
- 9bits indiquant l'adresse de base divisée par 32 (les adresses de base sont des multiples de 32 la division est donc exacte).
- 4bits donnant la taille du segment divisée par 64 (les segments ont forcément une taille multiple de 64 la division est donc exacte) ;

Cette manière de faire la segmentation permet une gestion de la mémoire offrant une protection et flexibilité tout à fait satisfaisantes pour ce type de processeur en ne consomant que 256o de la faible quantité de mémoire disponible.

Les multiplications par 32 et 64 sont très simples à réaliser en binaire simplement en ajoutant repectivement 5 et 6 zéros à la fin du nombre.

1. Quelle est la taille de l'adresse physique ?

2. Quelle est la taille maximale d'un segment ?

3. Quelle est la taille et la composition de l'adresse logique ?
Un extrait de la table des segments est donné ci-dessous ;

Segment	Processus	Base	Limite
C7	3	103	2
03	1	1CC	3
17	1	0E0	1
E5	2	000	A
06	7	0E5	4
2A	4	1B0	3

4. Donnez, pour le processus 3, les adresses physiques correspondant aux adresses logiques suivantes : `A8C7` et `1A2C`.

Exercice 5. Segmentation avancée On considère un système doté de 1Mo de mémoire physique utilisant la segmentation. Chaque processus peut utiliser un maximum de 256 segments d'une taille maximale de 64ko. 128 de ces segments sont locaux au processus et sont décris par une table des segments propre à ce processus. Les 128 autres segments sont commun à tous les processus et décris par une table des segment commune à tous les processus du système. Le bit de poids fort de l'adresse logique indique, quand il vaut 1, que le segment référencés par cette adresse logique est un segment global.

Un segment ne peut commencer qu'à une adresse multiple de 256, cela implique que le dernier octet de son adresse de base sera toujours 0 et qu'il n'est donc pas nécessaire de le stocker dans la table des segments.

Tous les segments, qu'ils soient locaux ou globaux, sont décris dans une des deux tables par leur adresse de base, leur limite ainsi que par 3bits de protection R, W et X, indiquant si un accès en lecture, en écriture ou en exécution est autorisé, ainsi qu'un bit S indiquant si l'accès au segment est réservé au mode superviseur du système.

1. Quelle est la taille des adresses physique ?

2. Quelle est la taille de l'adresse logique ?

3. Quelle est la taille de la table des segment locale d'un processus ?

On suppose qu'un processus P1 est présent dans le système, on donne ci-dessous des extraits des tables des segments. Vous pouvez considérer que toutes les lignes qui ne sont pas présentes sont enièrement à zéro. (la colonne RWXS indique la valeur des 4bits en notation binaire)

Table locale de P1

Seg.	Base	Limite	RWXS
00	000	0000	0000
0C	020	4000	1010
13	1D4	0800	1000
21	000	0400	0000
2A	C70	D704	1100
48	1D4	0800	1101

Table globale

Seg.	Base	Limite	RWXS
21	000	0200	1100
2A	A5C	0100	1100
6C	032	1000	0010

4. Que se passe-t-il lorsque le processus réalise des accès en écriture aux adresses suivantes :`2A04E5`, `A10386` et `0c1F5E` ?

5. Où est probablement stocké le code du processus P1 ?

Les question suivantes sont plus générales et sont destinées à vous amener à réfléchir sur les possibilitées offertes par ce type de système de gestion de la mémoire. La conception de ces systèmes est toujours un compromis entre les contraintes pratiques et les fonctionalitées offertes.

6. Qu'y a-t-il de particulier avec les segments locaux 13 et 48 ? À quoi sert ce type de configuration ?

7. Qu'y a-t-il de particulier avec le segment local 00 ? À quoi sert-il ?

8. En quoi ce système de gestion de la mémoire est-il particulièrement intéressant pour l'OS ?

9. Le fabriquant de ce système souhaite produire une nouvelle version disposant de 4Mo de mémoire, quelle(s) modification(s) proposeriez-vous pour supporter cette quantité de mémoire physique ?

Exercice 6. Fondements On suppose un espace d'adresses logiques de 8 pages de 1024 octets chacune permettant d'accéder à une mémoire physique de 32 cadres de pages.

1. Expliquez pourquoi les tailles de pages sont toujours une puissance de 2.

2. Combien de bits comporte l'adresse logique ? L'adresse physique ?

3. On suppose maintenant que le système dispose de 2048ko de mémoire logique organisé avec des pages de 8ko. Décrivez le système d'adressage logique.

4. Quelle est la taille maximum de la table des pages ?

5. On suppose que, dans le système de la question précédente, on a trois processus qui s'exécutent sur le système : P1 nécessistant 200ko, P2 de 545ko et P3 de 337ko. Quelle est la quantité de mémoire réellement utilisée par l'exécution de ces trois processus ? Quel est le taux de fragmentation ?

Exercice 7. Pagination à 1 niveau On considère un système 32bits utilisant la technique de pagination avec des cadres de pages de 64ko. Chaque processus peut utiliser au plus 4Go de mémoire virtuelle et le système peut accueillir jusqu'à 2048 processus.

1. Quelle est la taille maximale de la mémoire physique ?

2. Quelle est la taille de l'adresse logique ?

3. Quelle est la quantité maximale de mémoire virtuelle utilisée par le système ?

4. On suppose que le système dispose de 1Go de mémoire physique. Quelle est la taille de l'adresse physique ?

5. Sur combien de bits est codé le décalage dans l'adresse logique ? Dans l'adresse physique ?

6. Sur combien de bits est codé le numéro de page dans l'adresse logique ?

7. Sur combien de bits est codé le numéro de cadre dans l'adresse physique ?

8. Quelle est la taille maximale de la table des pages d'un processus ?

9. En considérant les huit premières entrées de la table de page donnée ci-après, calculez les adresses physiques correspondant aux adresses logiques 00030B72 et 00060D81 ?

Page	Cadre	Validité
0	000	0
1	010	0
2	000	1
3	0B3	1
4	2A0	0
5	09B	0
6	0F0	1
7	2DD	1

10. À l'inverse, pouvez-vous indiquer si la donnée située à l'adresse physique 2A0DE37 appartient au processus ? Expliquez.

11. Est-ce que 37 est une adresse physique valide ? Et 90003145 ? Expliquez.

Exercice 8. Pagination à 1 niveau On considère un système disposant de 4Mo de mémoire physique utilisant la technique de la pagination à 1 niveau avec des cadres de page de 1ko. Chaque processus dispose de 16Mo de mémoire virtuelle.

1. Quelle est la taille des adresses physiques ?

2. Sur combien de bits est codé le numéro de page dans l'adresse physique ?

3. Quelle est la taille des adresses logiques ?

4. Combien de lignes y a-t-il dans la table des pages d'un processus ?

Le tableau suivant donne un extrait de la table des pages d'un processus :

Page	Cadre	Validité
AC2	000	0
AC3	35E	1
AC4	12A	1
AC5	000	0
AC6	000	0
AC7	000	0
AC8	000	0
AC9	1A7	0

5. Calculez les adresses physiques correspondant aux adresses logiques 2B0E4C et 2B27F0.

6. Calculez l'adresse logique que le processus doit utiliser afin de réaliser un accès mémoire à l'adresse physique 4A8F1.

7. Que doit faire l'OS pour que le processus puisse réaliser un accès mémoire à l'adresse physique D12A ?

Exercice 9. Pagination à 2 niveaux On se place dans un système de mémoire de 4Go de mémoire géré de manière paginée avec des cadres de page de 4Ko. Chaque processus peut utiliser jusqu'à 64Mo de mémoire. Le système d'exploitation autorise jusqu'à 1024 processus.

1. Quelle est la taille (en bits) de l'adresse logique

2. Quelle est la taille (en bits) de l'adresse physique

3. Combien y a-t-il de cadres de page dans la RAM ?

4. Combien chaque processus peut-il contenir de pages ?

5. On suppose que la pagination se fait sur un seul niveau. Quelle quantité de mémoire est consommée par les tables des pages ?

6. On suppose que la pagination se fait sur deux niveaux avec un répertoire sur 6 bits. Quelle quantité de mémoire est consommée par un processus qui utilise les plages d'adresses logiques suivantes ?

de `0 00 1B 07` à `0 00 2C 08`
de `0 03 CA 00` à `0 0E 00 00`
de `0 0D AB 10` à `0 0F B1 CC`
de `0 2B 10 21` à `0 2B 3A 72`

Exercice 10. Pagination à 2 niveaux On considère un système de gestion de mémoire paginée à deux niveaux tel que :

- Les adresses virtuelles et physiques sont toutes deux codées sur 32bits ;
- Les 10 premiers bits de l'adresse virtuelle forment le premier index ($n_1 = 10$), les 10 bits suivant forment le deuxième index ($n_2 = 10$) et les 12 bits restant le déplacement ($m = 12$) ;
- Les tables de pages et le répertoire de chaque processus sont stockées dans la RAM et chaque entrée correspond à une adresse différente dans la RAM.

1. Sur combien d'octets seront codées les données du répertoire ? Quelle est la taille maximum du répertoire ?

2. Sur combien d'octets seront codées les données d'une table de page ? Quelle est la taille maximum d'une table de pages ?

3. Quelle est la quantité maximale de mémoire nécessaire pour stocker toutes ces tables ?

4. On considère un processus avec le répertoire suivant :

Index	Table
0	`0A7C 0002`
1	`0A7C 006E`
2	`0A7C 0072`
3	`0A7C 005A`

Et on souhaite résoudre l'adresse logique : `00C02DF6`. Quelle table est concernée ?

5. On suppose que la table des pages concernée (par le résultat de la question précédente) commence par les entrées suivantes :

Page	Cadre	Validité
0	`17AC0`	0
1	`02BCF`	1
2	`07F93`	1
3	`15BB0`	1

Donnez l'adresse physique correspondant à l'adresse logique `00C0 2DF6`.

Exercice 11. Segmentation paginée On considère un système muni de 64 ko de mémoire physique gérée de manière segmentée et paginée. Chaque processus peut utiliser 16 segments de 1 ko et le système supporte jusqu'à 256 processus. Les cadres de page font 512 o.

1. Quelle est la taille de l'adresse physique ?

2. Quelle est la taille de l'adresse logique ?

3. Quelle est la taille maximale de la mémoire virtuelle ?

4. Rappeler ce qu'est un segment global.

5. Dans cette question, on suppose que la moitié des segments sont globaux. Dans ce contexte, quelle est la taille de la mémoire virtuelle ?

6. Combien de pages un processus peut-il utiliser au maximum ?

7. On rappelle que l'adresse linéaire doit permettre d'adresser toute la mémoire d'un processus, mais qu'elle n'a pas besoin d'adresser l'ensemble de la mémoire virtuelle. Quelle est la taille et la composition de l'adresse linéaire ?

8. Quelle est la taille (en nombre de bits) de chaque ligne de la table des descripteurs lorsque tous les segments sont locaux ?

9. Même question si la moitié des segments sont globaux (ce qui est la configuration habituelle sur un OS moderne).

À un moment de l'exécution, plusieurs processus P_1, P_2, ..., P_N sont en exécution dans le système. L'état du système est partiellement décrit ci-après :

Table des segments de P_1 :

Segment	Limit	Base
00	085	0000
01	3B6	3000
02	341	2000
03	225	0086
04	05F	1B80

Table des segments de P_2 :

Segment	Limit	Base
00	0A5	1001
01	107	0C00
02	3B6	3000
03	0A3	01CF

Table des pages de P_1 :

Page	Cadre	Valide
00	36	0
01	7A	0
02	32	1
0E	00	1
0F	2A	0
10	2B	0
11	14	1
18	6C	1
19	55	1
1A	31	1
1B	30	0

Table des pages de P_2 :

Page	Cadre	Valide
00	24	0
01	32	0
08	2A	1
09	76	1
0A	54	0
18	6C	1
19	55	1
1A	31	1

10. Quelle est l'adresse physique qui correspondant à l'adresse logique 0B50 pour le processus P_1 ?

11. Quelle est l'adresse physique correspondant à l'adresse logique 0B50 pour le processus P_2 ?

12. Quelle est l'adresse physique correspondant à l'adresse logique 0C0D pour le processus P_1 ?

Exercice 12. Segmentation paginée On considère un système de gestion de mémoire de 64Mo de mémoire physique, gérée de manière segmentée et paginée avec deux niveaux de pagination. Un processus peut avoir au plus 256 segments. Chaque segment peut adresser au plus 64ko de mémoire. Enfin, la taille des cadres de page est fixée à 4ko et le répertoire contient 4 lignes.

1. Quelle est la taille et la composition de l'adresse logique ? Indiquez sa composition.
2. Quelle est la taille et la composition de l'adresse linéaire ?
3. Quelle est la taille et la composition de l'adresse physique ?

Soit un processus muni de la table des segments et du répertoire suivant :

Segments :

Segment	Base	Limite
00	BE 0A 75	05 00
01	BE 23 D1	00 BF
02	BE 00 DA	03 61
03	BE 1A 26	05 D2
04	BE 0F F0	00 CF

Répertoire :

Répertoire	Table	Valide
0	0	1
1	3	0
2	1	1
3	2	0

et de deux tables de pages (on ne fait figurer ici que quelques pages, les autres ont leur bit de validité à 0) :

Table 0 :

Page	Cadre	Valide
2A0	23 40	1
2A1	05 BB	1
2A2	00 00	0
2A3	14 E0	1

Table 1 :

Page	Cadre	Valide
3E0	00 00	0
3E1	27 FD	1
3E2	00 00	0
3E3	3A F6	1

4. Quelle est l'adresse physique correspondant à l'adresse logique 03 00 F0 ? Indiquez clairement les valeurs calculées pendant le processus de traduction.
5. Quelle est la quantité de mémoire physique utilisée par le processus ? Justifiez.

Exercice 13. Pagination Sv32 des processeur RISC-V L'architecture de processeur RISC-V propose différent modes de gestion de la mémoire en fonction des besoins de l'utilisateur. Lorsque les besoins en mémoire sont limités, le mode le plus simple Sv32 est utilisé. Dans ce mode les adresses logiques sont codées sur 32bits, les adresses physiques sur 34bits et une pagination à deux niveaux est utilisée avec des pages standard de 4ko. Le répertoire est les tables de pages contiennent le même nombre de lignes.

1. Quelle est la composition des adresses logiques ?
2. Quelle est la composition des adresses physiques ?

Le répertoire et la table des pages on la même structure, chaque ligne est composé d'un numéro de cadre sur 22bits et de 10bits d'informations supplémentaires parmis lesquels des bits de protection R, W et X indiquant si la ligne peut être utilisée pour un accès en lecture, écriture ou exécution. Pour le répertoire, ces trois bits sont normalement à zéro.

3. Quelle est la taille du répertoire et d'un table des pages ?
4. Il n'y a que 22bits disponibles dans le répertoire pour indiquer la position en mémoire physique de la table des pages. Pourquoi n'y en a-t-il pas 34 ?

Chaque ligne du répertoire dispose des bits R, W et X qui n'ont pas vraiment de sens si l'on considère qu'une ligne indique seulement ou se trouve la table de pages à consulter. Si par contre on considère que le répertoire découpe la mémoire logique en $2^{10} = 1024$ zones de 4Mo qui sont ensuite découpées en $2^{10} = 1024$ pages de 4ko par les tables de pages correspondantes, on peut améliorer la gestion de la mémoire d'une manière intéressante.

Une ligne du répertoire ou les trois bits R, W et X ont pour valeur 0, indiquent que le numéro de cadre pointe vers une table des pages gérant cette zone de 4Mo sous la forme de petites pages de 4ko. Une autre combinaison de ces trois bits indique que cette zone est gérée sous le forme d'une seule grosse page de 4Mo dont l'adresse physique est donnée par le numéro de cadre.

Cette technique permet d'allouer de gros blocs de mémoire contigüe sans avoir besoin d'utiliser de tables de pages ce qui économise de la mémoire.

5. Quelle quantité de mémoire est économisée par l'utilisation d'une "superpage" par rapport à une table de pages normale ?
6. Quel autre avantage présent l'utilisation de ces "superpages" ?

L'architecture RISC-V propose d'autres modes de gestion de la mémoire allant jusqu'à 57bits d'adresses logiques pour 66bits d'adresses logiques avec 5 niveaux de pagination. Chaque niveau peu indiquer soit une table de plus bas niveau, soit une superpage de grande taille. Pour ces modes permettant de gérer de très grandes quantité de mémoire, l'utilisation des superpages deviens critique pour gérer efficacement la mémoire.

Exercice 14. Gestion de la mémoire On se place dans un système de mémoire de 1Go de mémoire géré de manière paginée et segmentée avec des cadres de page de 4ko. Chaque processus peut utiliser jusqu'à 1024 segments. Chaque segment peut occuper 4Mo de mémoire. Le système d'exploitation autorise jusqu'à 1024 processus.

1. Quelle est la taille (en bits) de l'adresse logique ?

2. Quelle est la taille (en bits) de l'adresse physique ?

3. Combien y a-t-il de cadres de page dans la RAM ?

4. On suppose que l'OS utilise un mécanisme de mémoire virtuelle équitable : chaque processus dispose de la même proportion de mémoire physique. Complétez le tableau suivant en indiquant la probabilité de faire un défaut de page pour un processus, en fonction du nombre total de processus prêts, en exécution ou en attente dans le système et de la taille dudit processus. On suppose que toutes les pages du processus sont équiprobables.

Taille du processus →	512ko	16Mo	1Go
32 proc. présents			
512 proc. présents			
1024 proc. présents			

5. Pour réduire le nombre de défauts de page, on décide de passer à une politique d'allocation proportionnelle. On se place dans le cas 512 processus avec un nouveau processus de 16Mo. On suppose que les 511 processus précédents ont une moyenne de 2Mo. Indiquez la probabilité d'obtenir un défaut de page.

6. On suppose toujours que nous avons 1 processus de 16Mo et 511 processus de 2Mo. Si l'accès disque est 1000 fois plus lent que l'accès mémoire, que nous fixons à 1 unité de temps, donnez le temps moyen de traitement de 1000 accès mémoire par processus (512000 accès en tout) dans les deux cas suivants :

- Politique équitable ;
- Politique proportionnelle.

Qu'en concluez-vous ?

Exercice 15. Remplacement de page Durant son exécution, un programme accède successivement à la liste suivante de pages virtuelles de son espace d'adressage :

```
0, 1, 4, 2, 0, 1, 3, 0, 1, 4, 2, 3
```

On suppose que le système d'exploitation a alloué 3 cadres de pages au processus et que le cache est initialement vide.

1. Déroulez l'algorithme FIFO simple sur cet exemple et indiquez le nombre de défauts de pages.

2. Déroulez l'algorithme FIFO avec bit de seconde chance sur cet exemple et indiquez le nombre de défauts de pages.

3. Déroulez l'algorithme LRU sur cet exemple et indiquez le nombre de défauts de pages.

4. Quel serait le remplacement optimal (si on pouvait deviner à l'avance les appels que va faire le programme) et donc le taux de défaut de page minimum ?

Exercice 16. Remplacement de page On s'intéresse aux algorithmes de remplacement de pages dans un cache capable de contenir 4 pages. Le gestionnaire de mémoire accède successivement aux pages suivantes :

```
7, 1, 8, 2, 3, 1, 6, 1, 2, 5
```

Initialement, le cache est vide.

1. Déroulez l'algorithme FIFO simple sur cet exemple et indiquez le nombre de défauts de pages.

2. Déroulez l'algorithme FIFO avec bit de seconde chance sur cet exemple et indiquez le nombre de défauts de pages.

3. Déroulez l'algorithme LFU sur cet exemple et indiquez le nombre de défauts de pages.

4. Quel est le nombre de défauts de page minimal sur cet exemple ?

5. Quel serait le nombre de défauts de page minimal sur cet exemple avec un cache de taille 3 ?

Exercice 17. Remplacements de page On s'intéresse aux algorithmes de remplacement de pages dans un cache capable de contenir 5 pages. Le gestionnaire de mémoire accède successivement aux pages suivantes :

```
1, 7, 8, 2, 3, 1, 6, 1, 2, 7, 3, 5, 6
```

Initialement, le cache est vide.

1. Déroulez l'algorithme FIFO simple sur cet exemple et indiquez le nombre de défauts de pages.

2. Déroulez l'algorithme FIFO avec bit de seconde chance sur cet exemple et indiquez le nombre de défauts de pages.

3. Déroulez l'algorithme LRU sur cet exemple et indiquez le nombre de défauts de pages.

4. Quel est le nombre de défauts de page minimal sur cet exemple ? Justifiez.