TD Architecture des ordinateurs

Jean-Luc Dekeyser

Fiche 1 Nombres de l'informatique

Exercice 1

Une entreprise désire réaliser la sauvegarde de ses données sur un site distant. Le volume de données à sauvegarder est limité à 10Go/jour. La sauvegarde doit s'effectuer la nuit de 22h00 à 6h00. Les deux sites sont reliés par une ligne à 2Mbit/s.
On vous demande de vérifier si cette solution est réalisable et le cas échéant de proposer une solution qui permette cette sauvegarde.

Tableau de bit en octets

	bit	octet (byte)	Ko (KB)	Mo (MB)	Go (GB)	To (TB)
1 bit	1					
1 octet (byte)	8	1				
1 Ko (KB)	8 192	1024	1			
1 Mo (MB)	8 388 608	1 048 576	1024	1		
1 Go (GB)	8 589 934 592	1 073 741 824	1 048 576	1024	1	
1 To (TB)	8 796 093 022 208	1 099 511 627 776	1 073 741 824	1 048 576	1024	1

Exemple de lecture du tableau : 1 Mo = 1 048 576 octets

Exercice 2

- Question 1 : Combien de nombres peut-on coder sur 4, 8, 16 et 32 bits ? Quelles sont les bornes ?
- Convertir en binaire, puis en octal, et enfin en hexadécimal les nombres suivants : 100, 127, 128, 256, 1000, 1023, 1024, 10000

Exercice 3

1. Quel est le plus grand nombre que l'on peut stocker dans un octet ?
2. Quel est le plus grand nombre que l'on peut stocker dans deux octets ?
3. Quel est le plus grand nombre que l'on peut stocker dans quatre octets ?
4. Quel est le plus grand nombre que l'on peut stocker dans 64 bits ?
5. Combien d'octets faut-il pour stocker la chaîne << Que j'aime à faire apprendre... >>, sachant qu'en outre de chaque caractère de la chaîne elle-même, nous devons stocker sa longueur sous la forme d'un nombre entier codé sur quatre octets ?
6. Un disque dur a une capacité de 40 giga-octets, quel est le nombre de bits que l'on peut stocker sur ce disque ?
7. Un disque dur hautes performances vous est vendu comme pouvant contenir 72 giga-octets, or vous vous rendez compte qu'il ne peut contenir que 72 Milliards d'octets. Expliquez cette supercherie.
8. Le disque dur de la question précédente vous a été facturé 10000 Francs, de quelle somme vous avez été lésé ?

Fiche 2 Les portes logiques

Exercice 1

Considérer la fonction définie par la table de vérité ci-dessous :

A	B	C	$F(A,B,C)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

1 - Générer une expression logique correspondante (somme de produits et produit de sommes)
2 - Simplifier les deux expressions en utilisant les règles de l'algèbre de Boole.
3 - Construire le diagramme de Karnaugh et déterminer une expression logique associée.
4 - Considérer les fonctions logiques suivantes. Pour chacune d'elles,
– construire le diagramme de Karnaugh ;
– utiliser le diagramme pour simplifier les expressions.

(a) $F_1(A,B,C) = A \cdot \overline{B} \cdot C + A \cdot B \cdot \overline{C} + A \cdot B \cdot C$

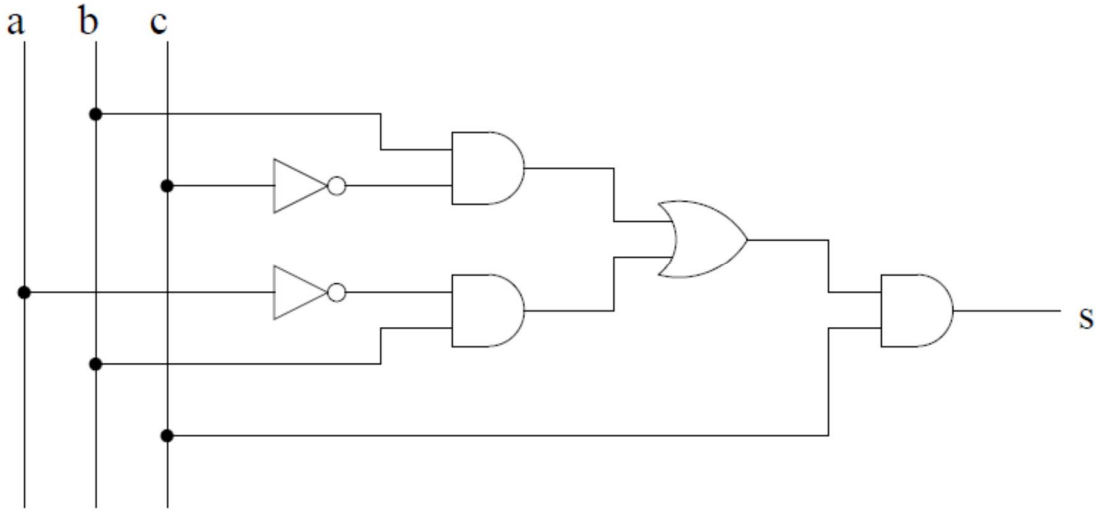
(b) $F_2(A,B,C) = \overline{A} \cdot \overline{B} \cdot \overline{C} + A \cdot \overline{B} + A \cdot B \cdot C$

(c) $F_3(A,B,C) = \overline{A} \cdot \overline{B} + \overline{A} \cdot B \cdot \overline{C} + \overline{B} \cdot \overline{C} + A \cdot \overline{B} \cdot C$

(d) $F_4(A,B,C,D) = B \cdot \overline{C} \cdot \overline{D} + \overline{A} \cdot B \cdot \overline{D} + A \cdot B \cdot C \cdot \overline{D}$

Exercice 2

1. Calculer la table de vérité du circuit logique suivant :

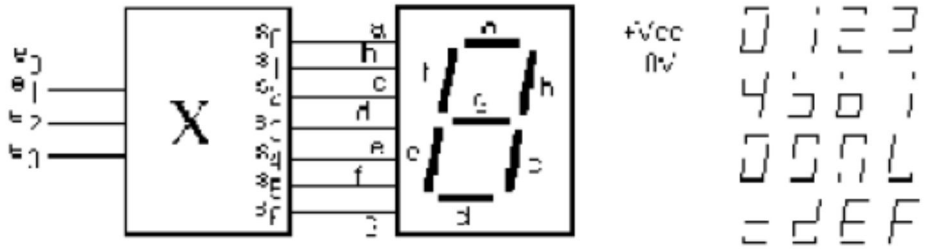


2. Donner une expression logique simple pour cette fonction logique.
3. Dessiner son circuit logique.

Fiche TD3 Circuits combinatoires

Exercice 1

On considère le circuit suivant :



Il est composé de deux parties :

- Un afficheur sept segments avec sept entrées logiques $a, b, \ldots, g$. **Un segment de l'afficheur est allumé si et seulement si l'entrée correspondante est à 0** ;
- Le circuit logique X que l'on doit réaliser. Il prend quatre entrées e_0, e_1, e_2, e_3 et doit donner en les sorties $s_0, s_1, \ldots, s_6$ telles que l'affichage soit le suivant :

e_3	e_2	e_1	e_0	affichage
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	A
1	0	1	1	b
1	1	0	0	c
1	1	0	1	d
1	1	1	0	E
1	1	1	1	F

- Trouver les tables de vérité de $s_0, s_1, \ldots s_6$.
- Trouver les expressions logiques pour chacune des sorties et dessiner le circuit complet.

Exercice 2

Montrer, en dessinant les circuits logiques, que toutes les fonctions logiques à deux variables peuvent être exprimées avec uniquement des AND, OR et NOT.

Exercice 3

- On rappelle qu'un système de fonctions logiques est dit complet s'il permet de calculer toutes les fonctions logiques.

1. Montrer que {AND, OR, NOT} est complet.
2. Montrer que {AND, NOT} et {OR, NOT} sont complets.

3. Montrer que {NAND} et {NOR} sont complets.

- Construire les fonctions de la question 1 avec des NAND. En sachant qu'un NAND prend deux transistors, calculer le coût correspondant si l'on n'utilise que des NAND.
Même question mais avec des NOR...

Exercice 1

Après avoir rappelé les tables de vérité des bascules D et JK, synchronisées sur front montant, donnez le chronogramme des sorties Q de chacune des bascules câblées ci- dessous en fonction d'une entrée d'horloge H.

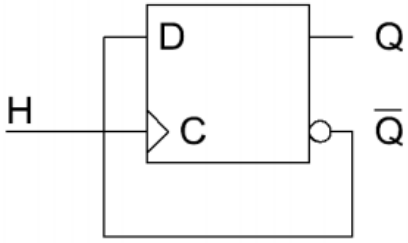


Figure 1

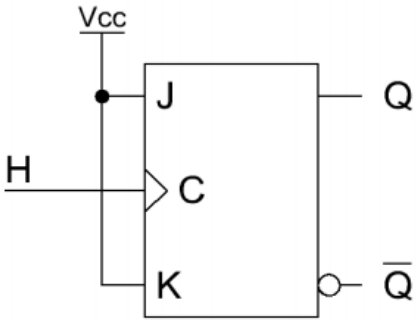
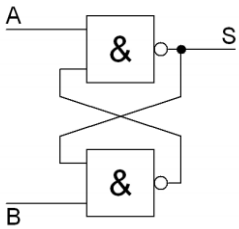


Figure 2

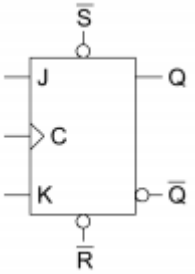
Exercice 2

1. Donnez la table de vérité du montage ci-dessous :
2. Quelle fonction reconnaissez-vous ?



Exercice 3

On dispose de bascules JK synchronisées sur front montant. Chaque bascule possède des entrées asynchrones prioritaires actives à l'état bas : set et reset.



1. Réalisez un compteur asynchrone modulo 16.
2. Modifiez le montage pour en faire un compteur modulo 12.
3. En partant de zéro, tracez son chronogramme sur un cycle complet.
4. Que suffit-il de faire pour remplacer les bascules JK par des bascules D ?

Fiche 5 Assembleur

Exercice 1

1) Donnez l'arbre dévaluation, la notation post-fixée et la notation préfixée de cette expression:

I = (A + B * C) / (D - E * F)

2) On dispose de 4 processeurs avec leur propre jeu d'instruction

0 adresse	1 adresse	2 adresses	3 adresses
PUSH M	LOAD M	MOV(X:=Y)	MOV(X:=Y)
POP M	STORE M	ADD(X:=X+Y)	ADD(X:=Y+Z)
ADD	ADD M	SUB(X:=X-Y)	SUB(X:=Y-Z)
SUB	SUB M	MUL(X:=X*Y)	MUL(X:=Y*Z)
MUL	MUL M	DIV(X:=X/Y)	DIV(X:=Y/Z)
DIV	DIV M		

M est une adresse mémoire (A, B, C, D, E, F, I). X, Y, Z sont soit une adresse mémoire, soit une adresse registre. La machine à 0 adresse utilise une pile. La machine à 1 adresse utilise un accumulateur. Les autres machines disposent de 16 registres R0.. R15. Chaque code opérateur est codé sur 8 bits. Une adresse mémoire est codée sur 16 Bits. Un numéro de registre est codé sur 4 bits.

- Donnez les 4 programmes réalisant l'expression de 1)
- Donnez la taille en bits de chaque programme

Exercice 2

Comme cela a été dit en cours, à chaque processeur est associé un langage machine. Voici un langage assembleur du langage machine du Motorola 68000 qui diffère de celui utilisé en TP (le X86)

La mémoire est une mémoire de mot de 8 Bits.
L'instruction MOVE S, D transfert la source S sur la destination D
Le $ signifie valeur en Hexa
Le # signifie adressage immédiat
Les registres Di, Ai sont des adressages direct registre, les registres sont de 32 bits
Les adresses mémoire sont des entiers.
Le ( ) signifie adressage indirect
le -( ) est un adressage pré-décrémenté (La valeur d'incrément est de 1 ou 2 suivant le nombre d'octets transférés)
Le ( ) + est un adressage post-incrémenté
Et le n(,) est un adresse indirect indexé avec déplacement
Les instructions manipulent des mots de 2 octets par défaut, de 1 octet si le « .B » est spécifié derrière l'instruction MOVE. Les bits de poids faible sont ceux qui sont modifiés.

Exécutez ce programme et pour chaque instruction montrez les changements intervenus sur les contenus des registres et de la mémoire. Les registres sont initialisés à $00000000.

```
MOVE      #$0010,D0
MOVE      #$10FB,$2000
MOVE      #$2002,A0
MOVE      -'(A0),D1
MOVE.B    (A0)+,D2
MOVE.B    (A0)+,D2
MOVE      #$8FFF,16(A0,D0)
```

Fiche 6 Programmation Assembleur

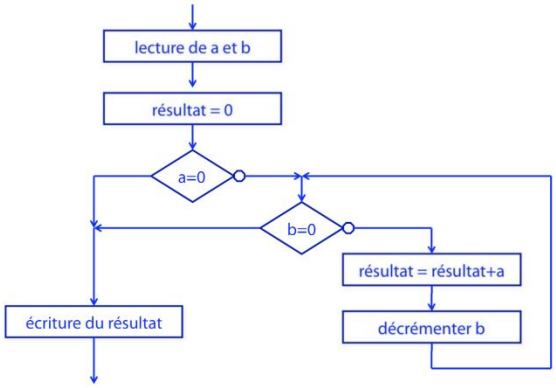
Exercice 1

Supposez que vous voulez multiplier deux variables a et b, stockées dans les positions de mémoire M[20] et M[21], respectivement, pour affecter cette valeur à la variable toto, stockée à la position de mémoire M[40] C'est-à-dire, vous voulez effectuer l'opération:
toto = a*b ou: M[40] = M[20]*M[21]

Le processeur possède 8 registres (R0...R7). Le registre R0 contient initialement la valeur 0. Les instructions du langage machine du processeur sont:

LOAD Rd, M[adr] Rd <=M[adr]
STORE M[adr], Rs M[adr] <= Rs
ADD Rd, Rs1, Rs2 Rd <= Rs1 + Rs2
SUB Rd, Rs1, Rs2 Rd <= Rs1 – Rs2
DEC R R <= R - 1
JMP adr sauter à adr
JZ Rs, adr si Rs = zero alors sauter à adr

Avant d'écrire le programme, nous devons trouver un algorithme réalisant la tâche voulue. Un algorithme possible serait:



1. Donnez le code correspondant à cet algorithme qui commence à l'adresse b'0000
2. Donnez le code correspondant à cet algorithme qui commence à l'adresse b'0010
3. Comment étendre le jeu d'instruction pour écrire un programme indépendant de l'adresse de départ. Réécrire le programme. Quelle différence avec l'utilisation d'un assembleur et d'un label.

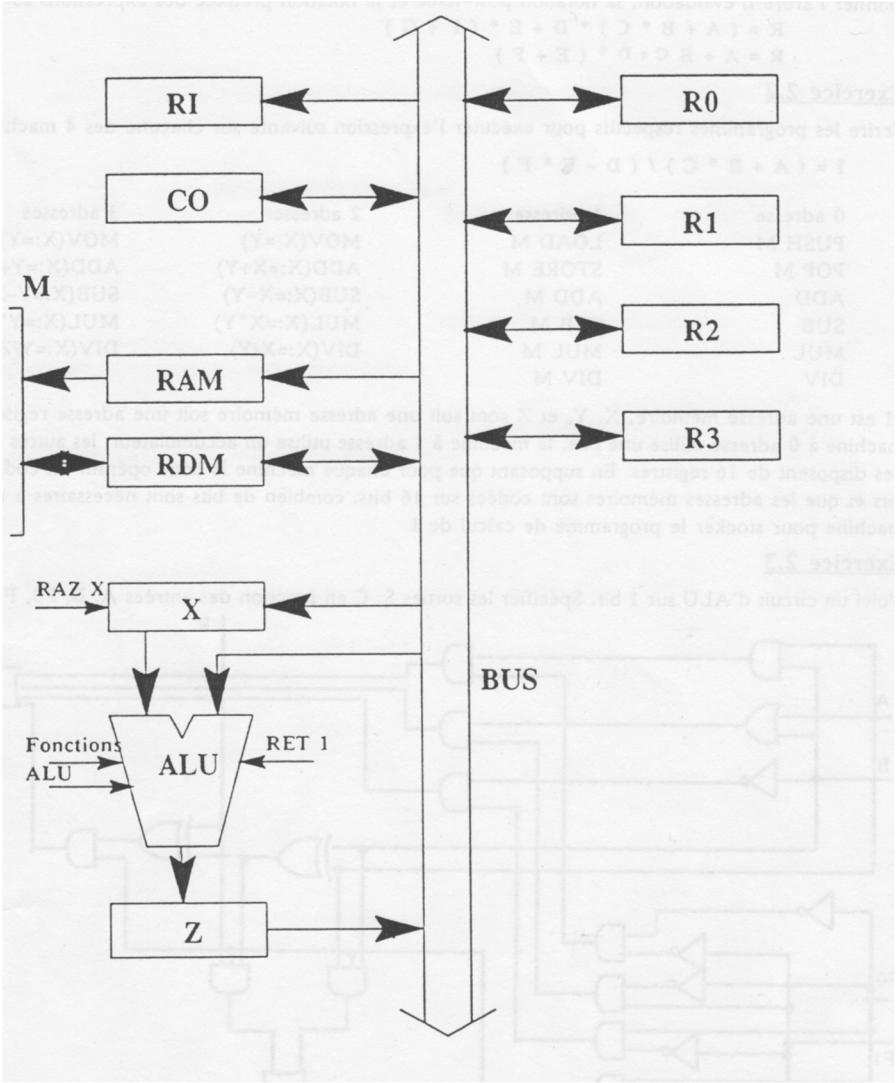
Exercice 2

Sachant que le bus d'adresse du processeur est de 16 bits avec un alignement à l'octet, quelle est la taille de l'espace mémoire maximum que celui-ci peut adresser ?

1. Quelles solutions existent pour adresser une plus grande zone mémoire ?
2. Où sont effectués les calculs ?
3. A quoi servent les registres suivants du processeur :
   i. PC/IP (ou CO/PI)
   ii. IR (ou RI)
   iii. SP (ou PP)
   iv. Accumulateur
4. Quelle tâche réalise le séquenceur dans un processeur ? l'ordonnanceur ?

Fiche 7 Micro programmation

Exercice 1



Ce processeur ne possède qu'un seul bus, il servira à transférer les données et les instructions. L'ALU peut exécuter 16 fonctions différents,,l'une d'entre elles se nomme ADD, elle ajoute X avec la valeur sur le BUS et evntuellement la retenue si celle-ci est positionnée. Ces fonctions se font en moins d'un cycle. Une entrée de l'ALU positionne la retenue à 1. Un mécanisme de remise à zéro du registre X est associé à un signal RAZX. Les registres RAM et RDM sont les buffers adresse et donnée vers/depuis la mémoire externe. L'accès à la mémoire est réalisé par deux signaux lecture et écriture, un seul cycle suffit pour obtenir le effcetuer l'accès. Les regsitres R0.. R3 sont les registres généraux.

1. Donnez la liste des micro-instructions de ce CPU et proposez une codification pas champs
2. Donnez les micro programme des phases d'exécution des instructions :

AA R1, R2	source , source/desta
ADD (R1) , R2	adressage indirect
ADD (R1) + , R2	adressage indirect postincrémenté

3. Donnez le micro programme de la phase de chargement

4. Reprendre les micro programme (2) avec un accès mémoire qui demande entre>2 et <3 cycles.

Fiche TD 8 La mémoire

Exercice 1

En général lorsqu'on ne dispose que de circuits ayant une certaine capacité mémoire et que l'on désire réaliser un espace mémoire de plus grande capacité nous devons faire face à deux problèmes différents :

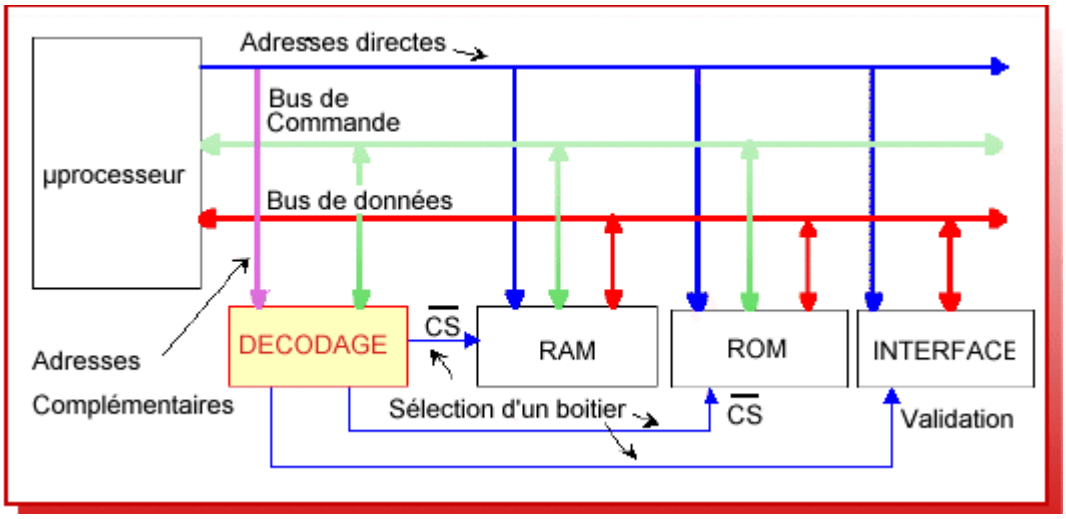
- étendre la largeur du bus de données ;
- étendre la largeur du bus d'adresse.

On dispose de deux ROM de taille 1024 mots de 8 Bits

1) Dessinez le circuit ROM
2) Proposez un cablage pour une mémoire de 1024 mots de 16 bit
3) Proposez un cablage pour une mémoire de 2048 mots de 8 bits

Exercice 2 Le décodage d'adresses

A une adresse présentée par le microprocesseur, un seul périphérique (RAM, ROM., etc.) doit répondre. Le processeur a 16 bits d'adresse.



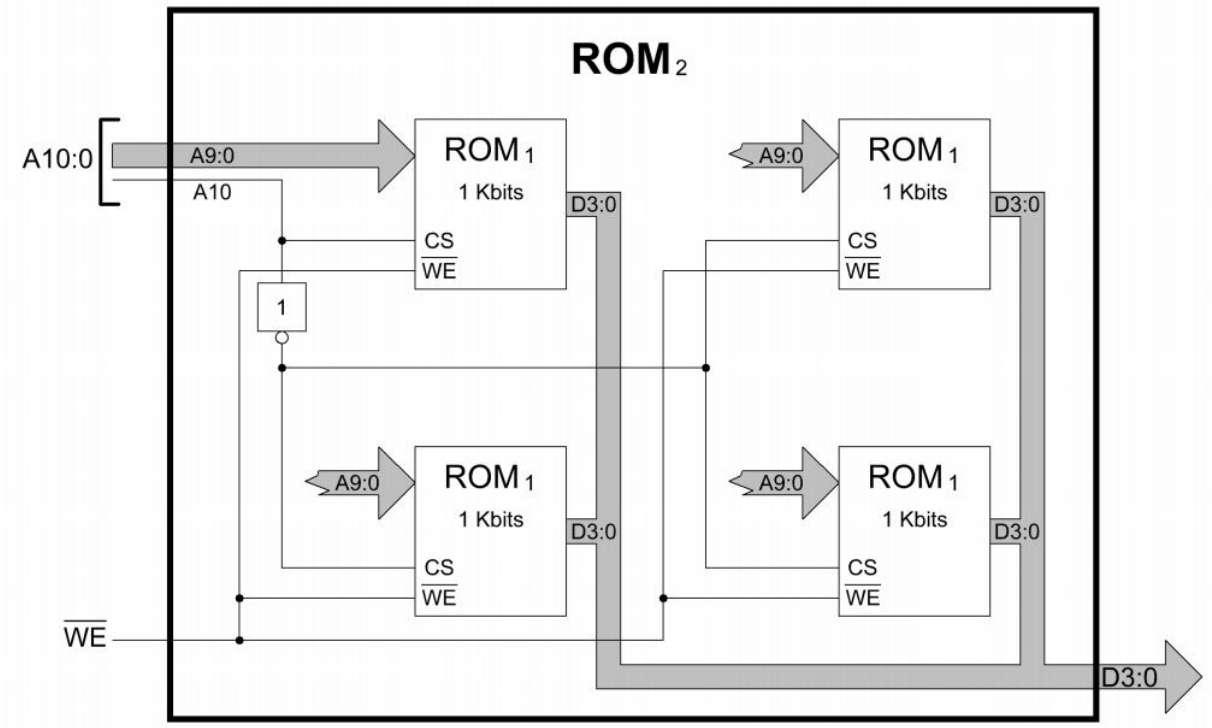
1) Le circuit mémoire RAM est implanté dans la zone adressable de $2000 à $2FFF. Proposez un décodage d'adresse câblé avec génération du Chip Select.
2) Placez la ROM en $3000-$30FF et l'interface en $FFFE-$FFFF
3) On voudrait réaliser un adressage programmable. Une commande matérielle permet de modifier l'espace adressable du circuit mémoire. On utilise pour cela 3 interrupteurs qui produisent suivant leur position un 1 ou un 0. Proposez un circuit qui réalise huit implantations possibles de la RAM en $0000, $2000...$E000.

Exercice 3 : adressage de mémoire

La capacité des circuits mémoire est en augmentation constante. Le nombre de broches nécessaire à l'adressage des mots mémoire suit cette augmentation, et il est souvent génant d'avoir un nombre de broches trop important sur un boîtier. Imaginez un moyen d'adresser 2^n mots mémoire en utilisant n' broches, avec $n' < n$. Faites un schéma et déterminez n'.

Exercice 4

Que semble faire le montage ci-dessous ? Trouvez les erreurs et proposez un montage qui fonctionne.



Fiche TD 9 Mémoire virtuelle

Exercice 1 Espace d'adressage paginé

Soit un système ayant 4 pages mémoire, la taille d'une page est égale à 100, l'adressage commence à 0. Un Programme P fait successivement référence aux adresses suivantes : 100, 210, 355, 120,420, 110, 200, 550, 139, 201, 395, 404, 505.

1) Donner la chaîne de références aux pages qui correspondent aux adresses.
2) Calculer le nombre de défauts de pages en appliquant la stratégie U.
3) Proposer une méthode qui donne un nombre minimum (optimal) de défauts de pages. Calculer le nombre de défauts de pages optimal.

Exercice 2

I) Soit un programme dont le code occupe 1024 octets en mémoire et qui utilise un vecteur avec 1000 éléments de type caractère (un caractère = un octet en mémoire). Ce programme est exécuté dans un système qui utilise la pagination de la mémoire dont la taille de la mémoire réelle est de 1 Mo, la taille d'une page est de 512 octets et les instructions à référence mémoire ont un champ d'adresse de 24 bits.

a) Donnez :

1) la taille de l'espace logique d'adressage
2) le nombre de bits du déplacement
3) le nombre de bits du numéro de page virtuelle
4) le nombre de bits d'une adresse réelle
5) le nombre de bits du numéro de page réelle (case)
6) le nombre d'entrées de la table des pages

b) Le chargement de ce programme en mémoire engendre-t-il une fragmentation interne de l'espace mémoire? Justifiez votre réponse.

II) On appelle "anomalie de Belady" le fait que le nombre de défauts de pages augmente quand on augmente le nombre de pages disponibles en mémoire physique. On appelle "algorithme à pile" un algorithme présentant la propriété d'inclusion :

M(m,r) inclus dans M(m+1, r),

où m est le nombre de pages physiques en mémoire et r un index dans le vecteur de références aux pages (ω).

M(m,r) représente l'ensemble des pages présentes en mémoire après r références mémoire, lorsque l'on dispose de m pages. Une condition suffisante pour qu'un algorithme ne présente pas l'anomalie de Belady est qu'il soit à pile.

a) Montrez l'anomalie de Belady pour l'algorithme FIFO avec m=3, puis m=4, pour le vecteur de références aux pages suivant : ω = { 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5 }.

b) Montrez sur cet exemple que l'algorithme FIFO n'est pas un algorithme à pile, en indiquant les états pour lesquels la propriété d'inclusion n'est pas vérifiée.

Exercice 3

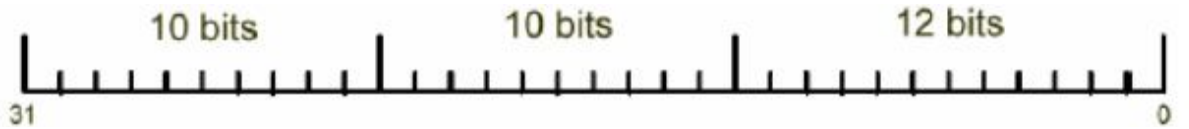
Considérez un système de gestion de mémoire qui a les caractéristiques suivantes :

- Un adressage virtuel sur 32 bits
- Une taille de Page de 4Ko
- Une mémoire physique de 1 Mo

a) Supposez que le système utilise la segmentation paginée et que l'adresse virtuelle est de la forme
:

Quelles sont les données manquantes à ce problème pour traduire l'adresse virtuelle de 32 bits suivante : (hexadécimal) AE854C9C en adresse physique ?
Dessinez un schémas qui utilise ces informations, et visualise les étapes à suivre pour effectuer cette translation. Combien y-a-t-il de pages par segment ?

b) Supposez que le système utilise une pagination à deux niveaux, où les entrées des tables de pages sont sur 4 octets. La première page contient des pointeurs vers les pages de deuxième niveau. La structure de l'adresse virtuelle est illustrée par la figure suivante :



b.1) Si un processus utilise tout l'espace adressable qui lui est fourni, combien de pages seront elles nécessaires pour contenir toutes les tables de pages de ce processus.
b.2) Un second processus nécessite 22Mo pour s'exécuter entièrement (son code, ses données, pile…). La partie contenant son code est disposée dans sa mémoire virtuelle aux adresses suivantes [2Mo à 6Mo-1], les données sont quant à elles dans l'intervalle [12Mo-21Mo-1].Si nous devons charger les tables de pages associées à ces deux parties, combien de pages de niveaux 2 seront chargées en mémoire centrale.

Fiche 10 Bus

Exercice 1 performance de bus

Le bus PC/AT, fonctionnant à une fréquence de 10 MHz, nécessite 4 cycles pour lire un mot mémoire.

1) Est-ce un bus synchrone ou asynchrone ? Justifiez votre réponse.

2) La taille d'un mot mémoire étant de 16 bits, calculez la bande passante de ce bus en bits/s et en octets/s.

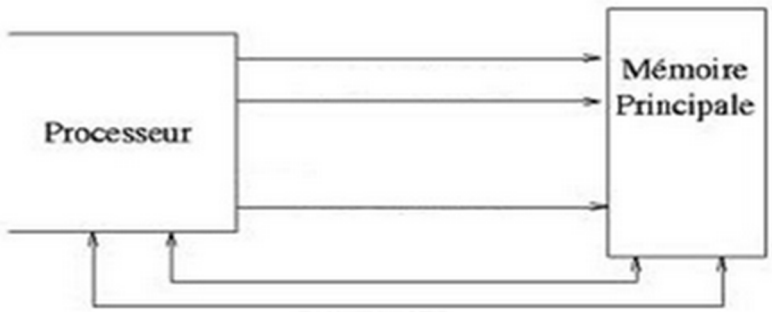
Exercice 2 débit de bus

1) Calculez la bande passante de bus nécessaire à un écran VGA couleur (640x480) pour afficher une vidéo à 50 et 60 images/seconde. Chaque pixel se voit attribuer une couleur parmi une palette fixe de 16 couleurs choisies parmi 262 144 possibles.

2) Soit un moniteur connectable par le bus USB sur un ordinateur. Le bus USB a un débit de 1,5 Mo/s. Avec un taux de rafraîchissement de 25 images/seconde et des images en noir et blanc, quelle résolution pourrait-on obtenir sur ce moniteur ?

Exercice 3

Quels sont les différents bus système et quel est le rôle de chacun d'entre eux?
Compléter le schéma ci-dessous :



Exercice 4

En utilisant des multiplexeurs, implémenter un bus permettant d'échanger des données entre 4 registres R0, R1, R2, R3 de 8 bits chacun. Le bus doit permettre de réaliser une opération de type Ri↔Rj, avec i<>j, c'est-à-dire l'échange des contenus de Ri et Rj en un seul cycle d'horloge

Exercice 1 Interruption Scrutation

Un dispositif d'entrées/sorties effectue dix requêtes par seconde, chacune d'entre elles nécessitant cinq mille instructions processeur pour être traitée.

1) Les entrées/sorties se font par interruption. Il faut mille instructions processeur pour sauvegarder le contexte et démarrer le gestionnaire de traitement de la requête, et de nouveau mille instructions processeur pour charger le contexte et revenir au processus principal. Combien d'instructions par seconde faut-il pour gérer les entrées/sorties ?

2) Les entrées/sorties se font maintenant par interrogation. Le processeur profite d'une interruption périodique préexistante tous les centièmes de seconde pour scruter le périphérique et voir s'il y a une requête à traiter. Il n'y a donc pas de coût supplémentaire de commutation de contexte. Chaque interrogation nécessite cinq cents instructions processeur, en plus du traitement de la requête si elle est présente. Combien d'instructions par seconde faut-il pour gérer les entrées/sorties ?

3) Quel peut être l'intérêt de gérer les entrées/sorties par interrogation plutôt que par interruption?

Exercice 2 DMA

Un disque dur peut transférer des données sur le bus à la vitesse de 8 Mo/s.

1) Les entrées/sorties se font par interrogation. Chaque scrutation du périphérique prend 0,1 µs, temps pendant lequel sont transférés 4 octets. Combien d'interrogations faut-il chaque seconde pour soutenir le débit du disque ? Quel pourcentage de temps processeur est utilisé pour cela ?

2) Les entrées/sorties se font maintenant par DMA. 1 µs est nécessaire pour initialiser le contrôleur DMA et encore 1 µs pour traiter l'interruption de fin. Le disque peut transférer 4 Ko à chaque session DMA. Combien de temps prend chaque session de transfert ? Quel pourcentage de temps processeur est utilisé pour gérer le transfert de données ?

Exercice 3

Le dialogue entre le CPU et le contrôleur (UART 8250) du port série (périphérique), s'effectue à travers des registres, et ce par des opérations d'écritures et de lectures successives dans ces derniers. On distingue essentiellement les registres suivants :

- Registre d'émission : Avant d'être émis, un caractère doit être chargé dans ce registre.
- Registre de réception : Ce registre joue le rôle d'intermédiaire entre l'UART et la CPU. La lecture des données reçues passe obligatoirement par la lecture de ce registre.
- Registre état de la ligne : Ce registre permet d'informer le CPU sur l'état de transfert de données. Si le bit 0 est à 1 ceci indique la fin de la réception d'un caractère et que ce dernier est disponible dans le registre buffer de réception, ce bit peut être remis à zéro par une opération de lecture du buffer de réception par le CPU ou par une opération d'écriture directe dans le registre état de la ligne. Le bit 5 est mis à « 1 » pour indiquer que le registre d'émission est vide et que l'UART est capable de recevoir un autre caractère dans ce buffer. On suppose que ces trois registres portent successivement les adresses : 3F8, 3F9 et 3FD en hexadécimal. On dispose de deux instructions, une d'entrée et une de sortie. IN et OUT

1. Ecrire un programme en pseudo-assembleur qui permet d'envoyer 3 caractères successifs vers le port série
2. Ecrire un programme en pseudo-assembleur qui permet de recevoir 2 caractères successifs depuis le port série.