II - Système à microprocesseur

1 Introduction

C'est en 1971 que le premier microprocesseur est sorti des laboratoires d'Intel (le 4004). Travaillant sur 4 bits et d'une puissance faible, l'intérêt de ce nouveau composant électronique ne fut pas évident jusqu'à ce que l'idée de le transformer en calculatrice fut trouvée.

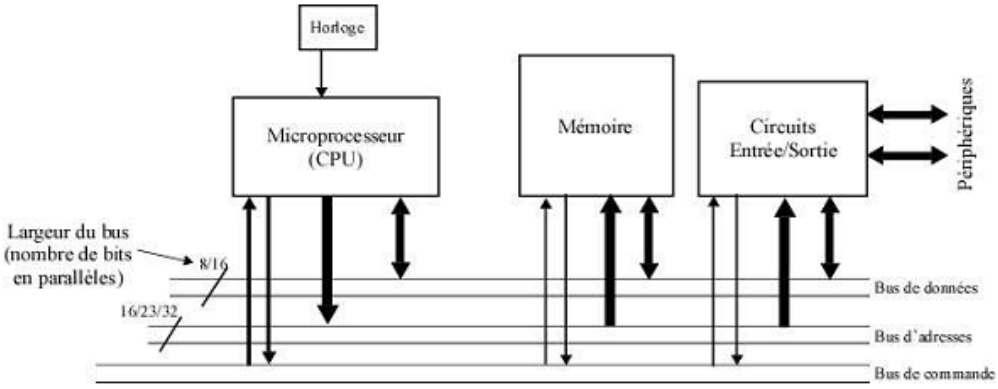
Sept ans plus tard, l'arrivée du 8088 multiplie déjà cette puissance de calcul par 200 ! Cette date correspond à la naissance des véritables micro-ordinateurs. Arrivent ensuite les microprocesseurs 68000 et 80286 (16 bits) avec les Macintosh et P.C. que nous connaissons. Ils ont introduisent l'image et le son.

Ensuite, tout n'est plus qu'une question de course à la puissance de calcul. Chaque bond technologique apporte sont innovation.

1.1 Structure générale

C'est un système logique programmables, conçu à base d'un *µp*, et comprend principalement :

- un µp;
- mémoires;
- interfaces.



1.2 Notion de HARDWARE (Matériel)

C'est tous les composants physiques qui constituent un système, à savoir circuits imprimés, composants électroniques, alimentation, châssis, …ça représente la partie palpable ou concrète du système.

- ***Microprocesseur*** :

C'est l'élément exécuteur (CPU, *Central Processing Unit)*, exécute les instructions qui se trouvent en mémoire (vive ou morte).

- ***Mémoire morte*** : elle contient des programmes et des données ineffaçables.
- ***Mémoire vive*** : elle contient des programmes et des données qui peuvent être effacées par le µp.
- ***Interfaces*** : circuits servants d'intermédiaire entre le µp et d'autres éléments qui communiquent avec ce circuit.
- ***Horloge*** : c'est la base de temps qui cadence le fonctionnement séquentiel du µp.

Le microprocesseur (CPU) échange des informations avec la mémoire et l'unité d'E/S, sous forme de mots binaires, au moyen d'un ensemble de connexions appelé bus. Un bus permet de transférer des données sous forme parallèle, c'est-à-dire en ; , faisant circuler n bits simultanément.

Les microprocesseurs peuvent être classés selon la longueur maximale des mots binaires qu'ils peuvent échanger avec la mémoire et les E/S : microprocesseurs 8 bits, 16 bits, 32 bits, ... Le bus peut être décomposé en trois bus distincts :

- le bus d'adresses permet au microprocesseur de spécifier l'adresse de la case mémoire à lire ou à écrire ;
- le bus de données permet les transferts entre le microprocesseur et la mémoire ou les E/S ;
- le bus de commande transmet les ordres de lecture et d'écriture de la mémoire et des E/S.

1.3 Notion de SOFTWARE (Logiciel)

C'est l'ensemble des programmes, des méthodes, et des règles de programmation destinées à faire fonctionner et à utiliser un systèmes programmable.

Le software est la partie abstraite liée à l'exploitation du système.

Catégories de Logiciel :

- ***Le programme de l'utilisateur*** :

C'est tout programme écrit en langage machine, en langage assembleur ou en langage évolué par l'utilisateur du système pour ses applications.

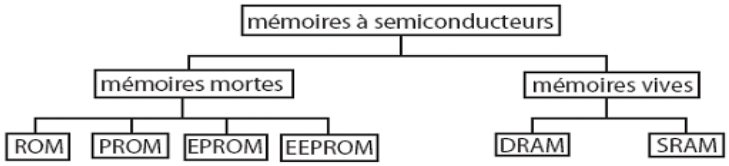
- ***Interpréteur et compilateur pour langage évolué*** :

Ce sont des programmes spéciaux qui interprètent les programmes de l'utilisateur écrits en langage évolué, en des programmes écrits en langage machine,donc compréhensible par le µp.

- ***Le programme système (système d'exploitation)***

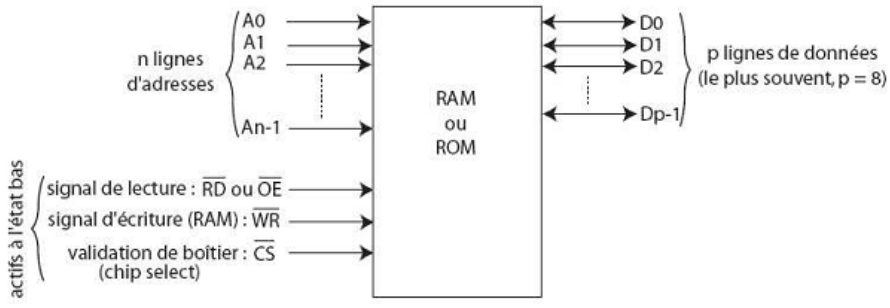
Le système d'exploitation (SE, en anglais Operating System ou OS) est un ensemble de programmes responsables de la liaison entre les ressources matérielles d'un ordinateur et les applications informatiques de l'utilisateur (traitement de texte, jeu vidéo…). Il fournit aux programmes applicatifs des points d'entrée génériques pour les périphériques.

2 Les mémoires à semi-conducteurs



On distingue deux types de mémoires :

2.1 Schéma fonctionnel d'une mémoire



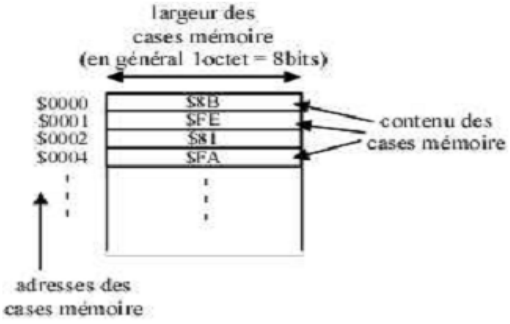
Le nombre de lignes d'adresses dépend de la capacité de la mémoire : ***n*** lignes d'adresses permettent d'adresser 2^n cases mémoire : ***8*** bits d'adresses permettent d'adresser 256 octets, 16 bits d'adresses permettent d'adresser 65536 octets (= 64 Ko), ...

Exp. :

Mémoire ***RAM 6264*** : 13 broches d'adresses de A0 à A12, 2^{13} = 8192 = 8 Ko (capacité = 8K×8 bits).

2.2 Organisation de la mémoire

La mémoire peut être vue comme un ensemble de cellules ou cases contenant chacune une information : une instruction ou une donnée. Chaque case mémoire est repérée par un numéro d'ordre unique : son adresse.



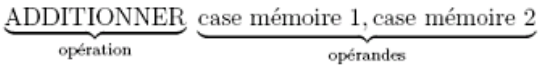
Une case mémoire peut être lue ou écrite par le microprocesseur (cas des mémoires vives) ou bien seulement lue (cas des mémoires mortes).

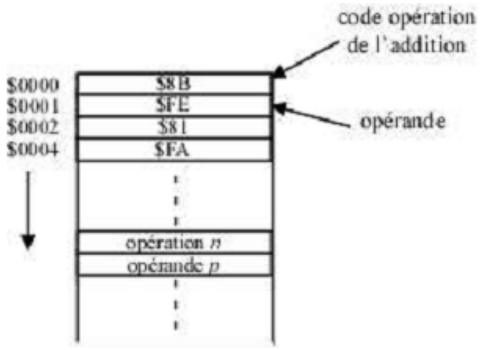
3 Fonctionnement d'un microprocesseur

Un microprocesseur exécute un programme. Le programme est une suite d'instructions stockées dans la mémoire. Une instruction peut être codée sur un ou plusieurs octets.

Format d'une instruction :

Exemple :





A la suite de chaque instruction, un registre du microprocesseur est actualisé en fonction du dernier résultat : c'est le registre d'état du microprocesseur. Chacun des bits du registre d'état est un indicateur d'état ou flag (drapeau).

Exp. : registre d'état du microprocesseur 68000

T	**-**	**S**	**-**	**-**	**I_2**	**I_1**	**I_0**	**-**	**-**	**-**	**X**	**N**	**Z**	**V**	**C**

Les indicateurs d'état sont activés lorsqu'une certaine condition est remplie, exemple : le ***flag* Z** est mis à 1 lorsque la dernière opération a donné un résultat nul, le ***flag* C** est mis à un lorsque le résultat d'une addition possède une retenue, ...

Les indicateurs d'état sont utilisés par les instructions de saut conditionnels : en fonction de l'état d'un (ou plusieurs) ***flags***, le programme se poursuit de manière différente.

Chaque instruction est caractérisée par le nombre de périodes d'horloge (ou microcycles) qu'elle utilise (donnée fournie par le fabricant du microprocesseur).

Exp. : horloge à 5 MHz, période **T = 1/f = 0, 2 µs**. Si l'instruction s'exécute en 3 microcycles, la durée d'exécution de l'instruction est : **3 × 0,2 = 0, 6 µs**.

L'horloge est constituée par un oscillateur `a quartz dont les circuits peuvent être internes ou externes au microprocesseur.

4 Types d'architecture d'un microprocesseur

4.1 Architecture CISC (Complex Instruction Set Computer)

C'est une architecture avec un grand nombre d'instructions. Le processeur doit exécuter des tâches complexes par instruction unique. Donc, pour une tâche donnée, une machine CISC exécute un petit nombre d'instructions mais chacun nécessite un pus grand nombre de cycles d'horloge (Intel 8086, Pentium…, Motorola 68000, PowerPC). Actuellement les deux technologies convergent : les processeurs CISC (Pentium par exemple) utilisent des instruction de plus en plus simples et exécutent parfois plusieurs instructions en un cycle d'horloge.

4.2 Architecture RISC (Reduced Instruction Set Computer)

Architecture dans la quelle les instructions sont en nombre réduit (chargement, branchement, appel sous-programme) et elle sont fréquemment utilisées. Le but est d'éliminer les instructions rarement employées et de consacrer les ressources matérielles à exécuter les instructions relativement simples en un cycle d'horloge et à émuler les autres instructions à l'aide des séquences basées sur les instructions élémentaires. On trouve donc une meilleure performance à une vitesse donnée (le gain en performance envisageable est important mais dépend de la qualité du compilateur). Processeurs RISC : PowerRISC (IBM/Motorola), SPARC (SUN), PA-RISC (HP).

4.3 Les processeurs spécialisés

4.3.1 Micro-contrôleurs

Ils contiennent un CPU, de la RAM, de la ROM, quelques ports d'E/S parallèles, des ports séries, des compteurs programmables (timers), des CAN/CNA, des interfaces pour réseaux de terrain ...

Ils sont en général utilisés pour contrôler des simples machines (appareils électroménagers, lecteurs de carte à puce...). Exemple de circuits :

- 80C186XX (80186, 16 bits, Intel)
- 68HC11, 68HC12 (6809, 8 bits, Motorola)
- 68HC16 (68000, 16 bits, Mototrola)
- PIC 16F84, 16F877 (Microchip)

4.3.2 Digital Signal Processor

Ce sont des processeurs dédiés aux traitements des signaux numériques. Une architecture particulière leur permet un traitement efficace des fonctions complexes telles que FFT, convolution, filtrage numérique ...

Exp. :

- TMS320 (Texas Instrument)
- 2100 et 21000 (Analog Device)
- 56000 (Motorola)

11.4. Exercice 4 : association de mémoires statiques asynchrones

Énoncé

Associer deux mémoires asynchrones de 2^n mots de m bits pour former une mémoire de 2^n mots de $2m$ bits, puis une mémoire de 2^{n+1} mots de m bits.

Solution

première association

Il faut d'abord remarquer que dans les deux cas, la mémoire obtenue a bien un nombre total de bits mémoire égal à la somme des bits mémoire des deux parties. Chaque boîtier initial possède $m \times 2^n$ bits mémoire, et leur somme fait donc $m \times 2^{n+1}$. Dans la première association, on aura 2^n mots de $2m$ bits soit $2 \times m \times 2^n = m \times 2^{n+1}$; dans la deuxième on aura aussi les $m \times 2^{n+1}$ bits.

Pour former une mémoire de 2^n mots de $2m$ bits, il faut mettre essentiellement toutes les lignes des deux boîtiers en commun, à l'exception des lignes de données qu'il faut mettre en parallèle. Cela donne le circuit de la figure IV.41.

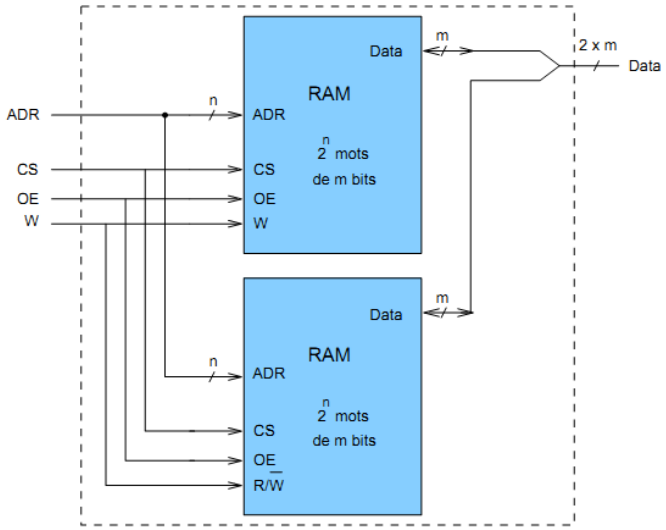


Figure IV.41. *Association de mémoires asynchrones, avec doublement de la largeur des données.*

Lorsqu'un accès mémoire (lecture ou écriture) est effectué, les deux boîtiers sont sélectionnés en même temps à la même adresse, et fournissent ou acceptent simultanément les deux moitiés de largeur m de la donnée globale de largeur $2m$.

deuxième association

Pour former une mémoire de 2^{n+1} mots de m bits, il va falloir mettre en commun les m lignes de données, et donc il ne va plus être possible de sélectionner les deux boîtiers en même temps, sous peine de court-circuit. Pour une moitié des adresses, il faudra sélectionner le premier boîtier, et l'autre pour l'autre moitié des adresses. La manière la plus simple est ici d'utiliser un des bits de l'adresse de $n + 1$ bits, par exemple adr[n], pour faire cette sélection (figure IV.42).

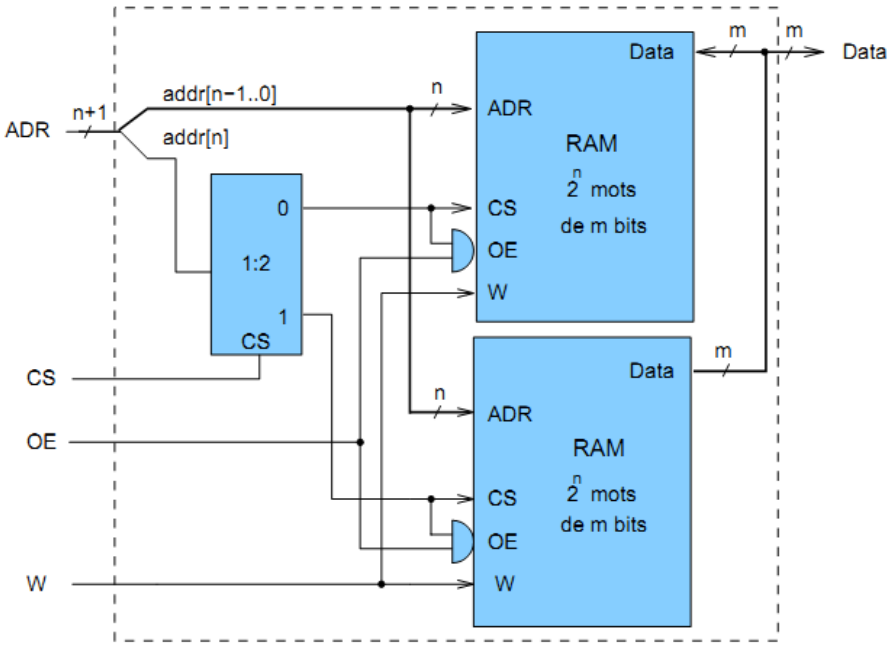


Figure IV.42. *Association de mémoires asynchrones, avec doublement du nombre d'adresses.*

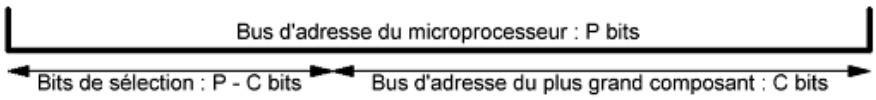
Ce bit est mis en entrée d'un décodeur 1 vers 2 : s'il vaut 0, c'est le boîtier mémoire du haut qui est sélectionné ; s'il vaut 1 c'est le boîtier du bas. Tout cela suppose que le signal *CS* global soit actif, car s'il ne l'est pas, le décodeur ne sélectionne aucun boîtier Le signal *OE* suit un traitement particulier : pour chaque boîtier, il ne doit être actif que si le *OE* global est actif, et si le boîtier est sélectionné ($CS = 1$). On est alors assuré qu'aucun court-circuit n'est possible, puisque les *OE* des deux boîtiers sont alors mutuellement exclusifs.

Dans cette configuration, l'espace des adresses de l'ensemble est $[0,\ 2^{n+1} - 1]$; la première moitié $[0,\ 2^{n} - 1]$ correspond au premier boîtier ; la seconde $[2^{n},\ 2^{n+1} - 1]$ correspond au second. Si on avait utilisé le bit de poids faible de l'adresse globale comme entrée du décodeur, la distribution des adresses entre les boîtiers aurait été différente : l'un aurait contenu les adresses paires, et l'autre les adresses impaires.

Exercice 1

Soit **P**, le nombre de bits d'adresse du microprocesseur, et **C**, le nombre de bits d'adresse du plus grand composant connecté au microprocesseur.

1. Quel est, en fonction de **P** et de **C**, le nombre total de composants que l'on peut connecter au microprocesseur avec un décodage linéaire ?



Le nombre de bits de sélection disponibles est **P – C**.

En décodage linéaire, on associe un bit de sélection par composant. **On peut donc connecter P – C composants au microprocesseur**.

2. Même question avec un décodage par zone.

En décodage par zone, on associe une combinaison de bits de sélection à un périphérique. **On peut donc connecter 2^{P-C} périphériques au microprocesseur**.

Exercice 2

On dispose d'une mémoire morte (ROM) de 8 Mib, d'une mémoire vive (RAM) de 8 Mib et de deux périphériques (P1 et P2) adressables respectivement sur 8 Kio et 4 Kio. On désire les rendre accessibles à un microprocesseur via les bus d'adresse (24 bits), de donnée (8 bits) et de commande. Les mémoires et les périphériques sont compatibles en largeur avec le microprocesseur. La ROM sera située dans les adresses les plus faibles, viendront ensuite la RAM, P1 et P2.

1. Donnez la taille du bus d'adresse de chaque mémoire et de chaque périphérique.

Afin de déterminer la taille des bus d'adresse, il faut commencer par déterminer la **profondeur** des différents composants. Leur largeur étant l'octet, c'est **le nombre d'octets** de chaque composant qu'il faut déterminer. Concernant P1 et P2, leur capacité est exprimée en octet. C'est donc leur profondeur qui nous est directement donnée.

- **ROM** : 8 Mib = (8 Mi / 8) octets = 1 Mio = 2^{20} octets → **20 fils d'adresse**
- **RAM** : 8 Mib = (8 Mi / 8) octets = 1 Mio = 2^{20} octets → **20 fils d'adresse**
- **P1** : 8 Kio = 2^{13} octets → **13 fils d'adresse**
- **P2** : 4 Kio = 2^{12} octets → **12 fils d'adresse**

Dans un premier temps, c'est le mode linéaire qui sera utilisé.

2. Quels bits d'adresse vont servir au décodage et à quel composant seront-ils associés ?

   Le décodage de type linéaire associe un composant à un bit de sélection du microprocesseur. Les bits de sélection sont les bits de poids fort. Nous avons ici quatre composants à relier au microprocesseur, ce seront donc les quatre bits de poids fort du bus d'adresse du microprocesseur (**A23** à **A20**) qui serviront au décodage.

   La ROM étant située dans les adresses les plus faibles, il faut lui associer le bit **A20**. Viennent ensuite la RAM et les deux périphériques que l'on associe respectivement aux bits **A21**, **A22** et **A23**.

Bit de sélection	Composant associé
A20	ROM
A21	RAM
A22	P1
A23	P2

← La ROM doit être située dans les adresses les plus faibles.

3. En tenant compte du signal **AS** (*Address Strobe*) que fournit le microprocesseur et qui indique si la valeur sur son bus d'adresse est valide, donnez la fonction de décodage ; c'est-à-dire les expressions du **CS** de chaque composant relié au microprocesseur.

   Tant que la valeur présente sur le bus d'adresse n'est pas valide (**AS** = 0), aucun composant ne doit être activé.

   - $\mathbf{CS_{ROM}}$ = AS.A20
   - $\mathbf{CS_{RAM}}$ = AS.A21
   - $\mathbf{CS_{P1}}$ = AS.A22
   - $\mathbf{CS_{P2}}$ = AS.A23

4. Donnez la représentation de l'espace mémoire avec toutes les adresses remarquables.

   Il faut déterminer l'adresse la plus basse et l'adresse la plus haute qu'occupe chaque mémoire et chaque périphérique. Il faut pour cela déterminer les bits de poids fort (sélection) et de poids faible (adressage du composant) du bus d'adresse du microprocesseur pour chaque composant. S'il existe des bits inutilisés, qui ne servent ni à l'adressage, ni à la sélection, ils seront positionnés à 0.

   Par exemple pour la ROM :
   - **A20** est à 1 pour la sélectionner.
   - **A21**, **A22** et **A23** sont à 0 pour désactiver les autres composants.
   - Pour son adresse basse, on positionne ses 20 bits d'adresse à 0.
   - Pour son adresse haute, on positionne ses 20 bits d'adresse à 1.

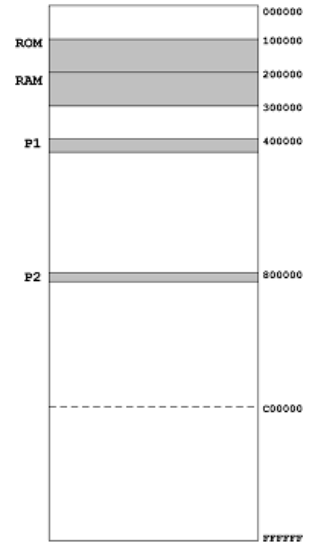
Ce qui donne pour tous les composants :

ROM basse : $0001\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 100000_{16}$
ROM haute : $0001\ 1111\ 1111\ 1111\ 1111\ 1111_2 = 1FFFFF_{16}$

RAM basse : $0010\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 200000_{16}$
RAM haute : $0010\ 1111\ 1111\ 1111\ 1111\ 1111_2 = 2FFFFF_{16}$

P1 basse : $0100\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 400000_{16}$
P1 haute : $0100\ 0000\ 0001\ 1111\ 1111\ 1111_2 = 401FFF_{16}$

P2 basse : $1000\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 800000_{16}$
P2 haute : $1000\ 0000\ 0000\ 1111\ 1111\ 1111_2 = 800FFF_{16}$



5. Quel est le principal défaut de ce type de décodage ?

   Le principal défaut de ce type de décodage tient au fait que plusieurs composants peuvent être activés en même temps. Cela peut entraîner un conflit d'accès sur le bus de donnée et causer des dommages.

6. Donnez les adresses interdites.

   Les adresses interdites sont celles qui créent des conflits. C'est-à-dire celles qui activent au moins deux composants en même temps. Ce sont donc toutes les adresses qui comportent au moins deux bits de sélection à 1.

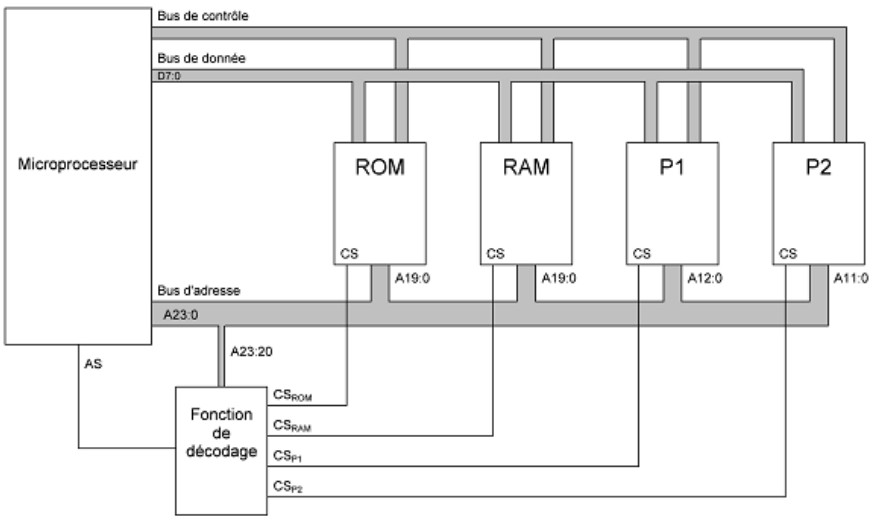
A23	A22	A21	A20	Composant
0	0	0	0	Aucun composant sélectionné
0	0	0	1	ROM seule
0	0	1	0	RAM seule
0	**0**	**1**	**1**	**Adresses interdites de 300000_{16} à $3FFFFF_{16}$**
0	1	0	0	P1 seul
0	**1**	**0**	**1**	**Adresses interdites de 500000_{16} à $7FFFFF_{16}$**
0	**1**	**1**	**0**	
0	**1**	**1**	**1**	
1	0	0	0	P2 seul
1	**0**	**0**	**1**	**Adresses interdites de 900000_{16} à $FFFFFF_{16}$**
1	**0**	**1**	**0**	
1	**0**	**1**	**1**	
1	**1**	**0**	**0**	
1	**1**	**0**	**1**	
1	**1**	**1**	**0**	
1	**1**	**1**	**1**	

7. Proposez une solution simple afin de supprimer les adresses interdites.

   Il faut modifier la fonction de décodage. L'entrée **CS** d'un composant doit être activée si son bit de sélection est à 1 et si les bits de sélection des autres composants sont à 0. Ainsi, aucun composant ne pourra être activé en même temps qu'un autre.

   - $\mathbf{CS_{ROM}} = AS.\overline{A23}.\overline{A22}.\overline{A21}.A20$
   - $\mathbf{CS_{RAM}} = AS.\overline{A23}.\overline{A22}.A21.\overline{A20}$
   - $\mathbf{CS_{P1}} = AS.\overline{A23}.A22.\overline{A21}.\overline{A20}$
   - $\mathbf{CS_{P2}} = AS.A23.\overline{A22}.\overline{A21}.\overline{A20}$

8. Donnez le schéma de câblage.



Pour la suite, on désire ajouter un périphérique P3 adressable sur 2 Kio (11 fils d'adresse).

9. Est-ce toujours possible en mode linéaire et pourquoi ?

   Le décodage de type linéaire associe un composant à un fil d'adresse du microprocesseur. Or ici, nous avons 5 composants, donc **5 fils du bus d'adresse sont nécessaires pour la sélection**. Les plus grands composants connectés au microprocesseur sont la ROM et la RAM avec 20 fils d'adresse. Sur les 24 fils du bus d'adresse du microprocesseur, il reste donc **4 fils disponibles** pour le décodage, **ce qui rend impossible l'utilisation du mode linéaire**.

On utilise maintenant le mode avec décodage (toujours avec P3).

On travaillera de préférence avec le moins de zones possible.

10. Quels bits d'adresse vont servir au décodage et à quelles combinaisons seront associés les différents composants ?

    Pour connecter 5 composants en utilisant le moins de zones possible, il faut utiliser 3 bits de sélection afin de découper l'espace mémoire en 8 zones. Ce seront les 3 bits de poids fort : **A23**, **A22** et **A21**.

A23	A22	A21	Composant associé
0	0	0	ROM
0	0	1	RAM
0	1	0	P1
0	1	1	P2
1	0	0	P3

← La ROM doit être située dans les adresses les plus faibles.

11. Donnez la nouvelle fonction de décodage.

- $\mathbf{CS_{ROM}} = AS.\overline{A23}.\overline{A22}.\overline{A21}$
- $\mathbf{CS_{RAM}} = AS.\overline{A23}.\overline{A22}.A21$
- $\mathbf{CS_{P1}} = AS.\overline{A23}.A22.\overline{A21}$
- $\mathbf{CS_{P2}} = AS.\overline{A23}.A22.A21$
- $\mathbf{CS_{P3}} = AS.A23.\overline{A22}.\overline{A21}$

12. Donnez la nouvelle représentation de l'espace mémoire avec toutes les adresses remarquables.

Il faut déterminer l'adresse la plus basse et l'adresse la plus haute qu'occupent chaque mémoire et chaque périphérique. Il faut pour cela déterminer les bits de poids fort (sélection) et de poids faible (adressage du composant) du bus d'adresse du microprocesseur pour chaque composant. S'il existe des bits inutilisés, qui ne servent ni à l'adressage, ni à la sélection, ils seront positionnés à 0.

Par exemple pour la ROM :

- **A21**, **A22** et **A23** sont à 0 pour la sélectionner (*cf.* tableau de la question 10).
- **A20** = 0 car il est inutilisé.
- Pour son adresse basse : on positionne ses 20 bits d'adresse à 0.
- Pour son adresse haute : on positionne ses 20 bits d'adresse à 1.

Ce qui nous donne pour tous les composants :

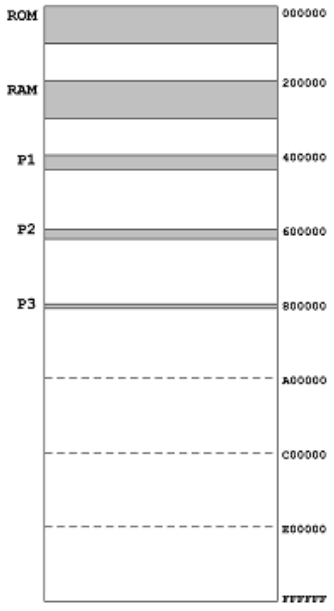
ROM basse : $0000\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 000000_{16}$
ROM haute : $0000\ 1111\ 1111\ 1111\ 1111\ 1111_2 = 0FFFFF_{16}$

RAM basse : $0010\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 200000_{16}$
RAM haute : $0010\ 1111\ 1111\ 1111\ 1111\ 1111_2 = 2FFFFF_{16}$

P1 basse : $0100\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 400000_{16}$
P1 haute : $0100\ 0000\ 0001\ 1111\ 1111\ 1111_2 = 401FFF_{16}$

P2 basse : $0110\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 600000_{16}$
P2 haute : $0110\ 0000\ 0000\ 1111\ 1111\ 1111_2 = 600FFF_{16}$

P3 basse : $1000\ 0000\ 0000\ 0000\ 0000\ 0000_2 = 800000_{16}$
P3 haute : $1000\ 0000\ 0000\ 0111\ 1111\ 1111_2 = 8007FF_{16}$



13. Quelle est la redondance des différents composants ?

La redondance est le nombre de combinaisons que l'on peut réaliser avec les fils inutilisés du bus d'adresse du microprocesseur. Les fils inutilisés sont ceux qui n'appartiennent ni aux fils de sélection, ni aux fils du bus d'adresse des composants.

Fils inutilisés = 24 fils en tout – 3 fils de sélection – fils d'adresse du composant

- **ROM** : 24 – 3 – 20 = 1 fil inutilisé → $\mathbf{2^1 = 2}$
- **RAM** : 24 – 3 – 20 = 1 fils inutilisé → $\mathbf{2^1 = 2}$
- **P1** : 24 – 3 – 13 = 8 fils inutilisés → $\mathbf{2^8 = 256}$
- **P2** : 24 – 3 – 12 = 9 fils inutilisés → $\mathbf{2^9 = 512}$
- **P3** : 24 – 3 – 11 = 10 fils inutilisés → $\mathbf{2^{10} = 1\ 024}$

15. Modifiez le schéma de câblage.