

$PC$ out ; $MAR$ in	}	$MDR = [PC]$
Lecture ; Attente		
$MDR$ out ; $MAR$ in	}	$MDR = [MDR] = [[PC]]$
Lecture ; Attente		
$MDR$ out ; $MAR$ in	}	$MDR = [MDR] = [[[PC]]]$
Lecture ; Attente		
$MDR$ out ; $Y$ in	}	$Y = [MDR] = [[[PC]]]$
$R_1$ out ; ADD ; $Z$ in		
$Z$ out ; $R_1$ in	}	$R_1 = [Y] + R_1$
$PC$ out ; INCRA ; $Z$ in ;		
$Z$ out ; $PC$ in	}	$PC = [PC] + 1$



TAB. 2 – Micro-instructions disponibles

Reg out	Sort le contenu du registre Reg sur le bus
Reg in	Met la valeur du bus dans le registre Reg
Lecture	Effectue une lecture de la mémoire à l'adresse MAR et met le résultat dans MDR
Écriture	Effectue une écriture dans la mémoire à l'adresse MAR de la valeur contenue dans MDR
Attente	Permet d'attendre la fin d'une lecture ou d'une écriture
ADD	Additionne les entrées A et B de l'ALU, met le résultat sur la sortie de l'ALU ( $F=A+B$ )
INCRA	Incrémente l'entrée A de l'ALU ( $F=A+1$ )
DECRA	Décrémente l'entrée A de l'ALU ( $F=A-1$ )
INCRB	Incrémente l'entrée B de l'ALU ( $F=B+1$ )
DECRB	Décrémente l'entrée B de l'ALU ( $F=B-1$ )
REPA	Reporte l'entrée A de l'ALU sur sa sortie ( $F=A$ )
REPB	Reporte son entrée B de l'ALU sur sa sortie ( $F=B$ )

## Exercice 2 : JSR et RTS

On considère une organisation à deux bus des chemins de données dans l'unité centrale de la Figure 2.

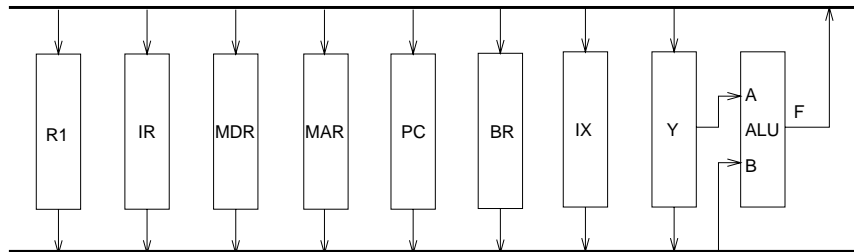


FIG. 2 – Organisation interne de l'unité centrale

Les abbréviations et les micro-instructions disponibles sont les mêmes que celles de l'exercice 1.

1. Écrivez la séquence des étapes de contrôle nécessaires à l'obtention et l'exécution de l'instruction JSR (Jump to Sub-Routine). Dans cette machine, l'instruction JSR occupe deux mots. Le premier mot est le code opération et le second mot contient l'adresse de départ du sous-programme. L'adresse de retour est sauvegardée dans une pile mémoire. Un pointeur de pile (SP) est

utilisé pour désigner en permanence le sommet de cette pile. On fera en sorte que  $[SP]+1$  pointe toujours sur l'élément du sommet de la pile (et  $[SP]+2$  sur le second élément de la pile, etc.).

Réponse \_\_\_\_\_

Exécution d'une instruction :

- (a) Prologue (aller chercher l'instruction et incrémenter le compteur de programme)<sup>2</sup> :

$PC$  out ;  $B$  in ;  $REPB$  ;  $MAR$  in

Lecture ;  $PC$  out ;  $INCRB$  ;  $PC$  in ; Attente

$MDR$  out ;  $REPB$  ;  $IR$  in

- (b) Décodage de l'instruction ;

- (c) Appel de la procédure de micro-code appropriée.

$PC$ out ; $INCRB$ ; $MDR$ in	}	$mem[SP] = [PC] + 1$
$SP$ out ; $REPB$ ; $MAR$		
Écriture ; Attente	}	$SP = [SP] - 1$
$SP$ out ; $DECRB$ ; $SP$ in		
$PC$ out ; $REPB$ ; $MAR$ in	}	$PC = [[PC]]$
Lecture ; Attente		
$MRD$ out ; $REPB$ ; $PC$ in		



2. Écrivez aussi l'instruction RTS (ReTurn from Sub-routine).

Réponse \_\_\_\_\_

$SP$ out ; $INCRB$ ; $SP$ in	}	$SP = [SP] + 1$
$SP$ out ; $REPB$ ; $MAR$ in		
Lecture ; Attente	}	$PC = [SP]$
$MDR$ out ; $REPB$ ; $PC$ in		



<sup>2</sup>Sur une ligne on représente les instructions qui peuvent être faites en parallèle

# TD d'architecture des ordinateurs IV

## Mémoire

### Exercice 1 : Conception d'une mémoire 8 bits

On désire construire une mémoire RAM de  $2M \times 8$  bits en utilisant des boîtiers mémoire  $256K \times 1$  bit.

- Quel est le nombre de fils d'adresse nécessaires pour adresser tous les octets de cette mémoire ?

Réponse \_\_\_\_\_

On veut adresser  $2 \text{ Mo} = 2 * 2^{20} = 2^{21}$  octets. Il faut donc 21 fils d'adresse.



- Rappeler l'intérêt d'utiliser des boîtiers  $256K \times 1$  bit plutôt que des boîtiers  $32K \times 8$  bits, qui ont la même capacité mémoire ?

Réponse \_\_\_\_\_

Pour un boîtier  $256K \times 1$  bit, on a :

- 1 broche de donnée ;
- 18 broches d'adresse ( $256K = 2^{18}$ ) ;
- $n$  broches classique ( $R/\overline{W}$ ,  $V_{cc}$ ,  $C_s$ ,  $Ground...$ ).

Soit  $19 + n$  broches.

Pour un boîtier  $32K \times 8$  bits, on a :

- 8 broche de donnée ;
- 15 broches d'adresse ( $32K = 2^{15}$ ) ;
- $n$  broches classique ( $R/\overline{W}$ ,  $V_{cc}$ ,  $C_s$ ,  $Ground...$ ).

Soit  $23 + n$  broches.

On préfère donc la première solution car génère un circuit moins complexe (moins de pistes).



- Dessiner schématiquement l'ensemble de la mémoire RAM construite à l'aide de ces boîtiers, en précisant leur nombre, et expliquer le fonctionnement du circuit sur un exemple.

Réponse \_\_\_\_\_

On veut faire une mémoire  $2M (= 8 \times 256K) \times 8 (= 8 \times 1)$  bits. On a donc besoin de  $8 \times 8 = 64$  boîtiers (8 bancs de 8 boîtiers = 8 bancs de  $256K$ ). On a des adresses sur 21 bits : les 3 premiers sont servant à sélectionner les bancs et les 18 autres sélectionne l'octet dans le banc).

Voir Figure-Réponse 1.



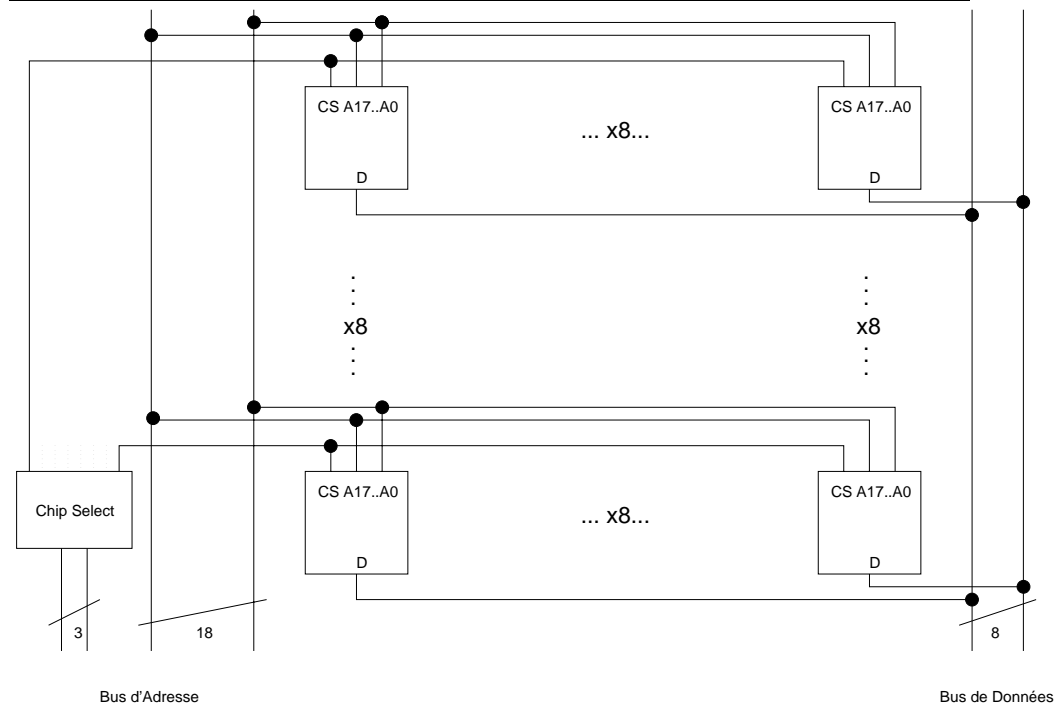
### Exercice 2 : Cartographie mémoire

Dessinez schématiquement la carte mémoire pour un ordinateur ayant un espace d'adressage de 16 Moctets et qui utilise :

- une RAM  $2M \times 8$  bits située à l'adresse  $0xB00000$  ;
- une ROM  $256K \times 8$  bits située à l'adresse  $0x000400$  ;
- un circuit d'interface série disposant de 4 registres 8 bits situé à l'adresse  $0x700000$  ;
- un circuit d'interface parallèle disposant de 32 registres 16 bits à l'adresse  $0x700100$ .

On précisera les adresses de fin des zones occupées. Représentez un circuit de décodage des adresses qui implémente cette carte mémoire. On pourra utiliser les différentes méthodes de décodage, ainsi que les différentes techniques pour l'implémentation. Justifier vos choix.

**FigRep 1** Mémoire 8 bits



### Réponse

Pour la cartographie, voir Figure-Réponse 2.

Le décodage de cette carte mémoire doit se faire par blocs successifs. Le premier étage du décodeur qui semble le mieux adapte, va diviser la mémoire en blocs de 256K à l'aide d'un selecteur sur les fils d'adresse  $A_{23}$  à  $A_{18}$ . Ce découpage permet d'isoler facilement la ROM et la RAM.

Or, la ROM est à cheval sur les deux premiers blocs. Quand le premier bloc est sélectionné ( $s_0 = 1$ ), si les fils  $A_{17}$  à  $A_{10}$  ne sont pas tous à 0, alors, il faut activer le boîtier ROM. Par contre, si c'est le second bloc ( $s_1 = 1$ ), il ne faut activer la ROM que si les fils  $A_{17}$  à  $A_{10}$  sont tous à 0. Les interfaces d'Entrées/Sorties sont quant à elles incluses dans le bloc 29 ( $s_{28} = 1$ ). Les fils d'adresse  $A_{17}$  à  $A_2$  tous à 0 active le circuit de l'interface série. Les fils  $A_{17}$  à  $A_9$  à 0,  $A_8$  à 1,  $A_7$  et  $A_6$  également à 0, active celui de l'interface parallèle.

La RAM recouvre entièrement 8 blocs de 256K :  $s_{44}$  à  $s_{51}$ . Il suffit de l'un de ces selecteurs à 1 pour activer la RAM (on utilise un OR).

**Remarque** : l'ordre physique (adresse du boîtier) des octets en mémoire ROM et RAM ne correspond pas à l'ordre logique (adresse du bus) !

Voir le schéma Figure-Réponse 3.

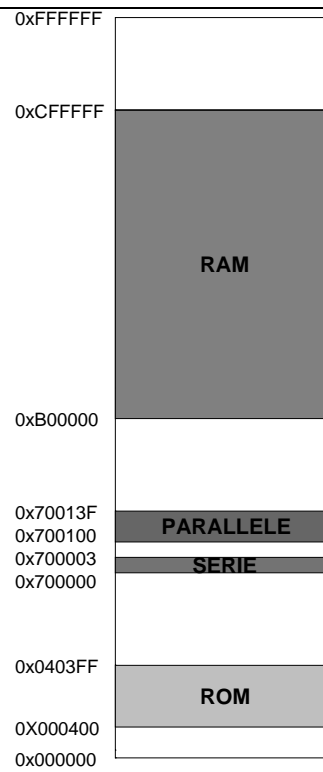


## Exercice 3 : Conception d'une mémoire 8/16 bits

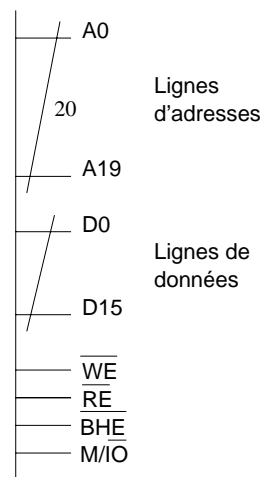
On dispose d'un processeur gérant 20 bits d'adresses et 16 bits de données. On veut que ce processeur puisse accéder à 4K d'octets de mémoire vive entre les adresses  $04000_H$  et  $04FFF_H$ . On utilise pour cela des boîtiers RAM (voir schéma fonctionnel Figure 1). On dispose aussi d'une PROM pour le décodage des adresses (cf. Figure 1).

On rappelle la signification des différentes lignes du bus du processeur.

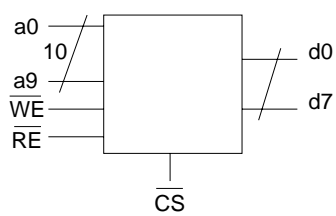
## FigRep 2 Cartographie mémoire



### Bus du processeur



### Schéma des boîtiers RAM



### Schéma du boîtier PROM

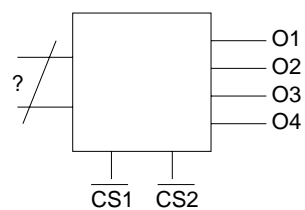
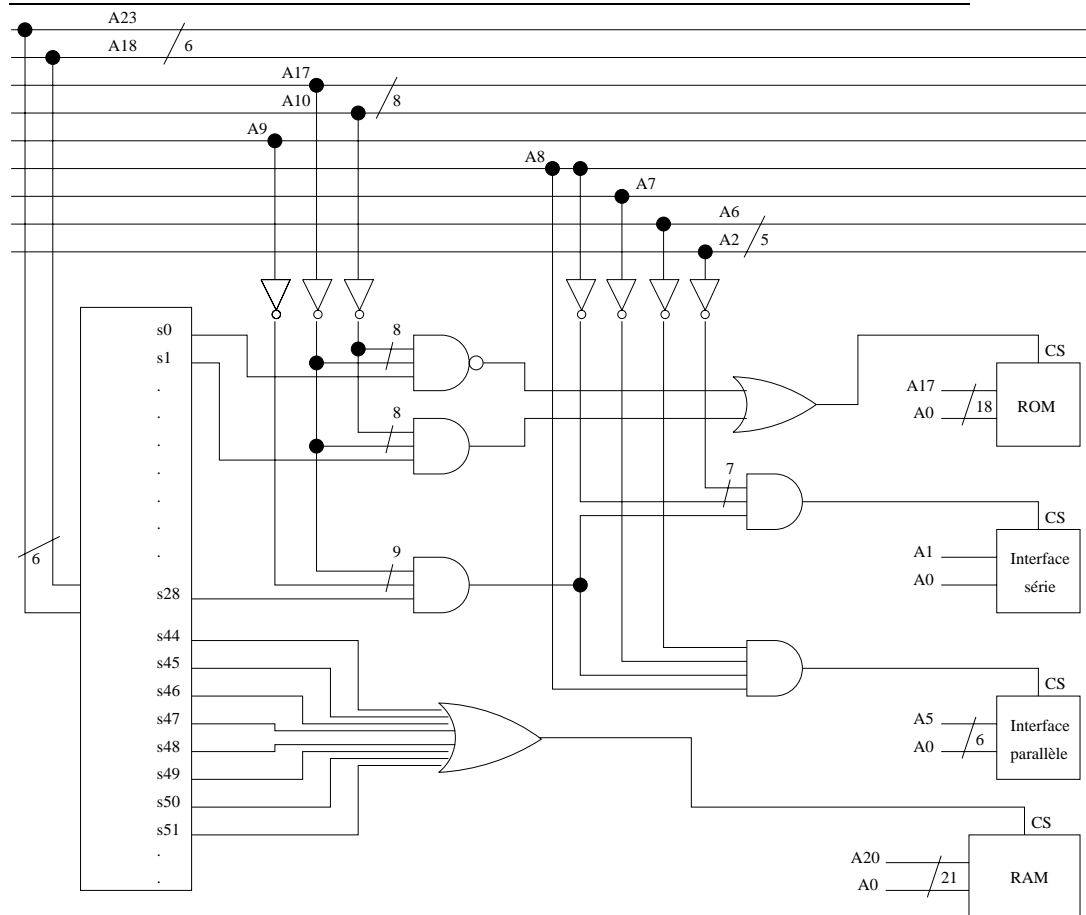


FIG. 1 – Schéma fonctionnel des boîtiers

**FigRep 3** Schéma du circuit de décodage



- $\overline{WE}$  (Write enable) signifie que le processeur demande une écriture dans la mémoire.
- $\overline{RE}$  (Read enable) signifie que le processeur demande une lecture dans la mémoire.
- $\overline{BHE}$  (Bus high enable) signifie que la partie haute du bus des données doit être utilisée (les 8 octets de poids forts).
- $M/\overline{IO}$  Memory or Input-Output indique si on demande un accès mémoire (ligne à 1) ou l'activation d'un port d'E/S (ligne à 0).

Les données peuvent transiter sur le bus sous forme d'octets ou de mots de 16 bits.

Les tableaux suivants donnent les différentes possibilités d'adressage et le comportement des boîtiers RAM.

Adresse	$\overline{BHE}$	Type de donnée
paire	0	mot de 16 bits
paire	1	octet
impaire	0	octet

Comportement	Condition
Lecture	$\overline{CS} = 0$ et $\overline{RE} = 0$
Ecriture	$\overline{CS} = 0$ et $\overline{WE} = 0$
Non sélectionné	$\overline{CS} = 1$

La PROM est activée si  $\overline{CS1} = 0$  et  $\overline{CS2} = 0$ . Répondez aux questions suivantes :

- Combien de boîtiers RAM sont-ils nécessaires ?

Réponse \_\_\_\_\_  
On utilise des boîtiers possédant 10 bits d'adresse. On peut adresser  $2^{10}$  octets = 1 Koctets. On veut une mémoire de 4Ko il faut donc 4 boîtiers.



- À quoi sont reliées les entrées-sorties  $d_0$  à  $d_7$  des différents boîtiers RAM ?

Réponse \_\_\_\_\_  
On veut pouvoir adresser des mots de 8 bits et des mots de 16 bits. On ne peut adresser que les mots de 16 bits qui sont situés sur des adresses paires. Les bits de poids forts des mots de 16 bits (adresses impaires) sont envoyés sur les bits  $D_{15}$  à  $D_8$  du bus de données (les bits de poids faibles sur les bits  $D_7$  à  $D_0$ ). On va diviser la mémoire en 2 bancs (de 2 boîtiers chacun) : un banc pour les adresses paires et un banc pour les adresses impaires. Les entrées-sorties  $d_7$  à  $d_0$  des boîtiers du banc des adresses impaires seront reliées aux bits  $D_{15}$  à  $D_8$  du bus de données. Les entrées-sorties  $d_7$  à  $d_0$  des boîtiers du banc des adresses paires seront reliées aux bits  $D_7$  à  $D_0$  du bus de données.



- À quoi sont reliés les fils d'adresses  $a_0$  à  $a_9$  des différents boîtiers RAM ?

Réponse \_\_\_\_\_  
Gestion des zones mémoire selon les boîtiers :  
– Boîtier 1 : Adresses paires entre  $0x04000$  et  $0x047FF$  ;  
– Boîtier 2 : Adresses impaires entre  $0x04000$  et  $0x047FF$  ;  
– Boîtier 3 : Adresses paires entre  $0x04800$  et  $0x04FFF$  ;  
– Boîtier 4 : Adresses impaires entre  $0x04800$  et  $0x04FFF$ .

	$A_{19}$ à $A_{12}$	$a_{11}$	$A_{10}$ à $A_1$	$A_0$
$0x04000$	0000 0100	0	000 0000 000	0
$0x047FF$	0000 0100	0	111 1111 111	1
$0x04800$	0000 0100	1	000 0000 000	0
$0x04FFF$	0000 0100	1	111 1111 111	1

Les bits  $A_{19}$  à  $A_{12}$  servent à sélectionner la zone mémoire  $0x04XXX$  ( $\overline{A_{19}} \cdot \overline{A_{18}} \cdot \overline{A_{17}} \cdot \overline{A_{16}} \cdot \overline{A_{15}} \cdot A_{14} \cdot \overline{A_{13}} \cdot \overline{A_{12}}$ ).

Le bit  $A_{11}$  sert à sélectionner les zones  $[0x04000 : 0x047FF]$  (boîtiers 1 et 2) et  $[0x04800 : 0x04FFF]$  (boîtiers 3 et 4).

Le bit  $A_0$  sert à sélectionner les adresses paires et impaires : si  $A_0=0$ , on sélectionne les boîtiers 1 et 3 sinon on sélectionne les boîtiers 2 et 4.

Les bits  $A_{10}$  à  $A_1$  servent à sélectionner les adresses dans les boîtiers. Ils sont donc reliés aux fils d'adresses  $a_9$  à  $a_0$  des boîtiers.

4. À quoi sont reliés les commandes  $\overline{CS}$  des boîtiers RAM ?

Réponse \_\_\_\_\_

Les commandes  $\overline{CS}$  des boîtiers RAM sont reliés au mécanisme de sélection de boîtiers à savoir les sorties  $O_1$ ,  $O_2$ ,  $O_3$  et  $O_4$  du boîtier PROM.

5. Donnez, en fonction des adresses  $A_0$  à  $A_{19}$  et des lignes  $M/\overline{IO}$  et  $\overline{BHE}$ , l'expression booléenne des sorties de la PROM. A quoi sont reliées les entrées de la PROM ?

Réponse \_\_\_\_\_

Le bit  $M/\overline{IO}$  doit être à 1 (sélectionne l'accès mémoire).

Le bit  $\overline{BHE}$  sélectionne les types d'accès.

$A_{11}$	$A_0$	$\overline{BHE}$	$O_1$	$O_2$	$O_3$	$O_4$	
0	0	0	1	1	0	0	accès sur 16 bits
0	0	1	1	0	0	0	accès sur 8 bits (adresse paire)
0	1	0	0	1	0	0	accès sur 8 bits (adresse impaire)
0	1	1	0	0	0	0	impossible
1	0	0	0	0	1	1	accès sur 16 bits
1	0	1	0	0	1	0	accès sur 8 bits (adresse paire)
1	1	0	0	0	0	1	accès sur 8 bits (adresse impaire)
1	1	1	0	0	0	0	impossible

On obtient les expressions (attention actif à l'état bas) :

$$\begin{aligned}
 - O_1 &= (\overline{A_{19}} \cdot \overline{A_{18}} \cdot \overline{A_{17}} \cdot \overline{A_{16}} \cdot \overline{A_{15}} \cdot A_{14} \cdot \overline{A_{13}} \cdot \overline{A_{12}} \cdot M/\overline{IO}) \cdot (\overline{A_{11}} \cdot \overline{A_0}) \cdot (\overline{CS1} \cdot \overline{CS2}) ; \\
 - O_2 &= (\overline{A_{19}} \cdot \overline{A_{18}} \cdot \overline{A_{17}} \cdot \overline{A_{16}} \cdot \overline{A_{15}} \cdot A_{14} \cdot \overline{A_{13}} \cdot \overline{A_{12}} \cdot M/\overline{IO}) \cdot (\overline{A_{11}} \cdot \overline{BHE}) \cdot (\overline{CS1} \cdot \overline{CS2}) ; \\
 - O_3 &= (\overline{A_{19}} \cdot \overline{A_{18}} \cdot \overline{A_{17}} \cdot \overline{A_{16}} \cdot \overline{A_{15}} \cdot A_{14} \cdot \overline{A_{13}} \cdot \overline{A_{12}} \cdot M/\overline{IO}) \cdot (A_{11} \cdot \overline{A_0}) \cdot (\overline{CS1} \cdot \overline{CS2}) ; \\
 - O_4 &= (\overline{A_{19}} \cdot \overline{A_{18}} \cdot \overline{A_{17}} \cdot \overline{A_{16}} \cdot \overline{A_{15}} \cdot A_{14} \cdot \overline{A_{13}} \cdot \overline{A_{12}} \cdot M/\overline{IO}) \cdot (A_{11} \cdot \overline{BHE}) \cdot (\overline{CS1} \cdot \overline{CS2}) ;
 \end{aligned}$$

6. Faites un schéma global du montage.

Réponse \_\_\_\_\_

Voir Figure-Réponse 4.

## Exercice 4 : Mémoire 6 bits

Toujours à partir de boîtiers  $2K \times 4$  bits, on désire maintenant construire une mémoire  $4K \times 6$  bits.

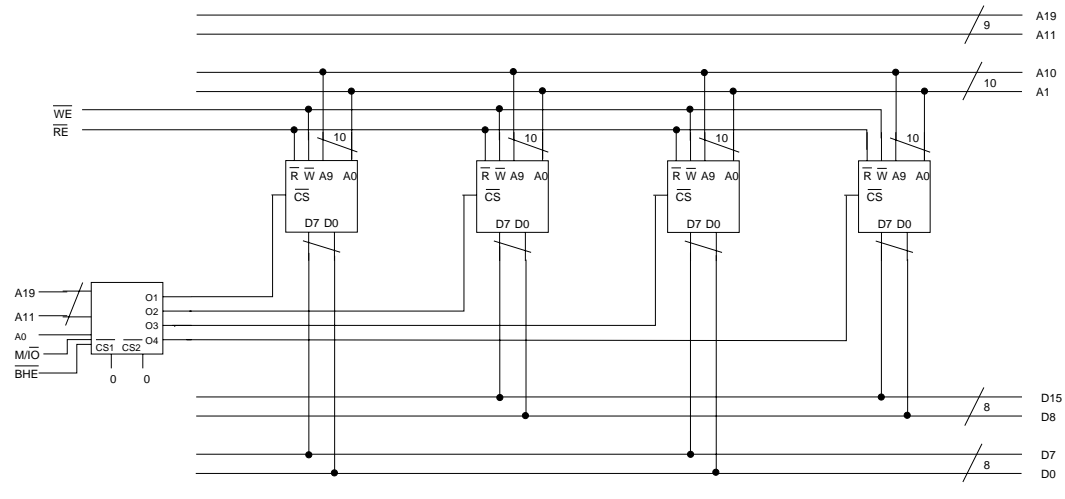
- Quel nombre minimal de boîtiers peut-on utiliser ? Proposer une solution qui utilise ce nombre minimal de boîtiers. On pourra donner les équations logiques associées aux broches des boîtiers et aux lignes du bus de données.

Réponse \_\_\_\_\_

Il faut 3 boîtiers  $2K \times 4$  bits pour obtenir la capacité mémoire nécessaire à une mémoire  $4K \times 6$  bits, donc c'est le nombre minimal de boîtiers nécessaires. On



**FigRep 4** Schéma global



peut utiliser 4 boîtiers, comme pour obtenir une mémoire 8 bits et sans utiliser les deux bits de poids fort. Mais cette solution fait perdre de la mémoire. Une méthode permet d'utiliser le nombre minimal de boîtiers. Elle consiste à aligner les trois boîtiers et d'utiliser les deux bits de poids faible de chaque boîtier pour les adresses paires et les deux bits de poids fort pour les adresses impaires. Les equations logiques associées aux broches du boîtiers sont :

**Notation :**  $x_y(n)$  signifie l'entrée  $x_y$  du boîtier  $n$ .

$$A_1 = a_0(x) \dots A_{11} = a_{10}(x)$$

$$D_0 = d_0(0) \cdot \overline{A_0} + d_2(0) \cdot A_0$$

$$D_1 = d_1(0) \cdot \overline{A_0} + d_3(0) \cdot A_0$$

$$D_2 = d_0(1) \cdot \overline{A_0} + d_2(1) \cdot A_0$$

$$D_3 = d_1(1) \cdot \overline{A_0} + d_3(1) \cdot A_0$$

$$D_4 = d_0(2) \cdot \overline{A_0} + d_2(2) \cdot A_0$$

$$D_5 = d_1(2) \cdot \overline{A_0} + d_3(2) \cdot A_0$$



- Représentez schématiquement cette mémoire avec tous les circuits nécessaires à son fonctionnement.

Réponse \_\_\_\_\_

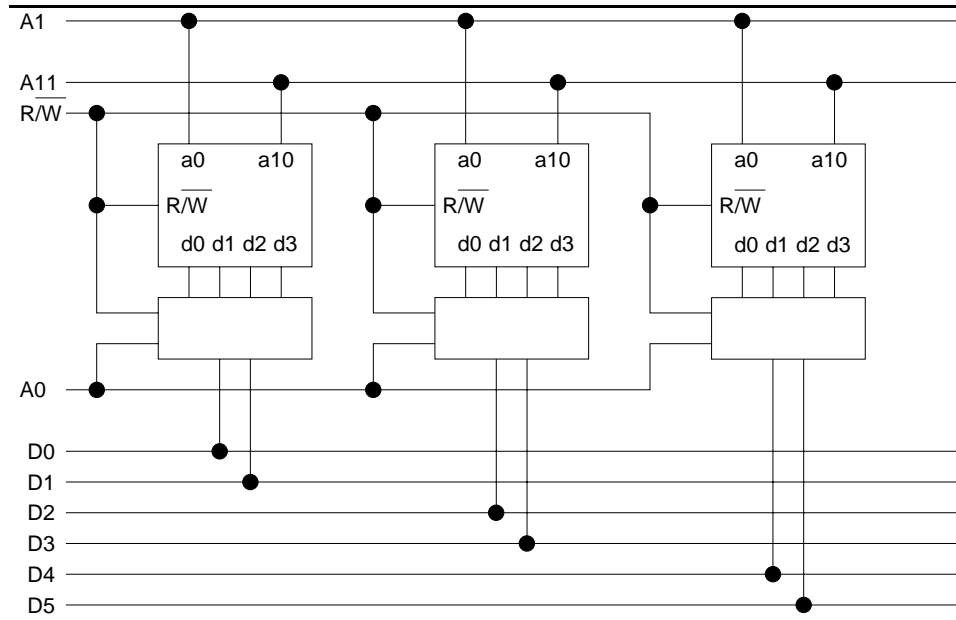
Pour sélectionner les poids forts ou faibles des mémoires, on peut utiliser un multiplexeur (voir Figure-Réponse 6). Mais cette configuration permet seulement la lecture de la mémoire.

Pour effectuer une écriture, on ne peut pas mettre une valeur sur seulement deux bits sur quatre. Les deux autres bits seraient perdus. Il faut d'abord copier le mot de 4 bits de chaque boîtier dans un registre, puis modifier le registre par le multiplexeur, et enfin recopier le mot entier dans la mémoire. Le registre doit permettre de ne modifier que certains bits. On peut utiliser 4 bascules JK indépendantes ou 2 registres 2 bits.

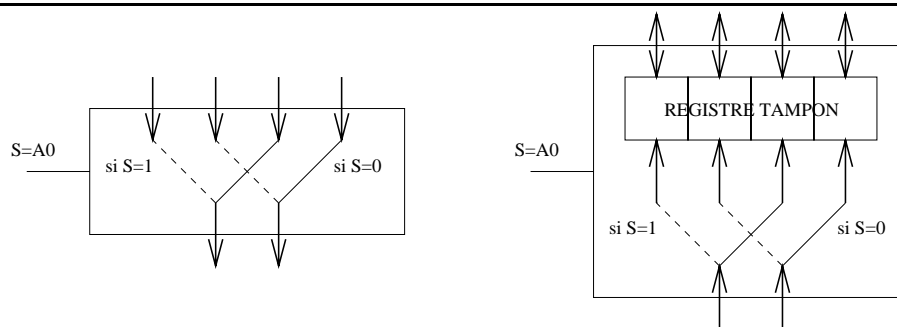
Voir le schéma global en Figure-Réponse 5 et le détail de multiplexeur en Figure-Réponse 6.



**FigRep 5** Mémoire 6 bits



**FigRep 6** Détail du multiplexeur



(a) Multiplexeur en lecture seulement

(b) Multiplexeur en écriture seulement