GIF-3002 Mémoires, Matériel

Ce document présente les mémoires de systèmes microprocesseurs en général. Il définit d'abord ce qu'est une mémoire ainsi que plusieurs mots reliés aux mémoires. Ensuite, le principe d'opération de plusieurs mémoires est présenté. Finalement, ce document traite d'accès à la mémoire.

1 Introduction (Rappel)

La mémoire d'un système microprocesseur contient des données ou des instructions, sous forme de bits. Ces bits sont regroupés en mots et chacun des mots de mémoire est à emplacement de la mémoire qui peut être accédé en utilisant une adresse. Chaque adresse désigne donc un ensemble de bits (un mot) de la mémoire, un peu comme l'index d'un tableau.

	Mémoire de 64K*8							
Adresse	b7	b6	b5	b4	b3	b2	b1	b0
0x0000	1	0	0	0	0	0	0	0
0x0001	0	1	1	1	1	0	1	0
0x0002	0	0	1	0	1	0	1	1
0x0003	0	1	1	1	1	0	0	0
0x0004	1	0	1	0	0	0	1	1
0x0005	0	0	0	0	1	0	0	0
0x0006	1	1	0	0	0	0	0	0
0x0007	0	0	1	0	0	0	1	1
0x0008	0	0	0	0	0	1	0	0
0x0009	0	0	0	1	0	1	0	0
0x000A	0	0	0	0	0	1	0	1
0x000B	0	0	0	0	1	0	0	1
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…
0xFFFE	1	1	1	1	1	0	1	0
0xFFFF	0	1	0	1	1	1	1	1
	Taille des mots = 8 bits							

Figure 1 - Memoire 64K*8

Deux chiffres servent à décrire la capacité d'une mémoire, c'est-à-dire la quantité de données qu'elle peut contenir : la taille du mot et le nombre d'adresses/mots. Le nombre d'adresses multiplié par la taille des mots, en bits, donne la capacité, en bits, de la mémoire.

Dans l'exemple ci-dessus, la mémoire a 64K adresse et 8 bits par adresse (le mot de la mémoire est un octet). Elle a une capacité de 512Kbits. Par ailleurs, on retrouve la valeur 0x05 à l'adresse 0x000A…

1.1 Types de mémoire

Il y a plusieurs sortes de mémoire. Grosso-Modo, elles se regroupent en deux catégories : les mémoires volatiles, qui perdent leur contenu lorsqu'elles ne sont plus alimentées, et les mémoires non volatiles qui gardent leur contenu même lorsqu'elles ne sont plus alimentées.

Certaines mémoires volatiles doivent être accéder régulièrement afin de na pas perdre leur contenu. Ces variables sont dites dynamiques : il faut les rafraîchir régulièrement. Les autres mémoires, par opposition, sont statiques.

> *Qualifier de dynamique une mémoire requérant un système de rafraichissement pour fonctionner correctement est un bon coup de marketing!*

Les mémoires qui ne s'écrivent qu'une seule fois sont dites ROM (Read Only Memory) ou OTP (One Time Programming). Les autres mémoires sont RAM (Random Access Memory).

Les mémoires qui peuvent s'effacer (et être réécrites) à l'aide d'une procédure spéciale ont des noms qui leurs sont propres. Flash et EEPROM s'effacent électriquement mais en appliquant une tension prédéterminée sur la mémoire, UVPROM nécessite de l'ultra-violet...

La table suivante résume les caractéristiques de quelques types de mémoire communs :

Noms	Volatile	Dynam.	Read only	Coût*	Vitesse
SRAM (bascule D)	oui	non	non	cher	très rapide
DRAM (trans + cond)	oui	oui	non	moyen	rapide
ROM (burned fuses)	non	non	oui	faible	moyen **
EPROM = UVPROM	non	non	o, eff. UV	moyen	moyen **
EEPROM	non	non	o, eff. EL.	moyen	moyen **
Flash	non	non	o, eff. EL.	cher	moyen **
Surface magnétique	non	non	non	faible	très lent

* Les coûts sont très relatifs. Exemple: le coût de la flash décroît à tous les 2 mois!
** La vitesse dépend de plusieurs facteurs.

La vitesse d'accès de la SRAM est autour de 2ns à 10ns par mot. Celle de la DRAM, entre 10ns et 50ns. Enfin, la vitesse d'accès de la FLASH est autour de 10-20ns en lecture, mais de quelques centaines de millisecondes en écriture… À titre comparatif, lire un bloc de données (~256 octets) d'un disque dur sans cache prend autour de 10ms.

1.2 Taille du mot de mémoire et alignement de mémoire

Les mémoires ont habituellement des mots de 8 bits, 16 bits, 32 bits ou 64 bits, mais n'importe quel nombre de bits de données est possible. Le plus souvent, on retrouve 8 bits d'information à une adresse spécifique de la mémoire.

Dans les microprocesseurs et les entrées/sorties, comme dans la mémoire, les données sont souvent regroupées en mots de N octets afin de faciliter la gestion. Le bus de données permet généralement de véhiculer un ou plusieurs mots.

Par ailleurs, un accès aux données est dit aligné lorsque l'adresse utilisée est un multiple de la taille des mots. Lors d'un accès aligné, tous les octets lus ou écrits sont dans le même mot. Lors d'un accès non-aligné aux données, l'adresse utilisée n'est pas un multiple de la taille des mots : les octets lus ou écrits appartiennent bien souvent à des mots différents.

1.3 Principe de fonctionnement général

De manière générale, tous les types de mémoire ont un bus de données de la taille du mot de mémoire. Elles ont aussi généralement un bus d'adresse dont la taille est le log2 du nombre de mots. Enfin, toutes les mémoires ont des lignes de contrôles permettant de lire la mémoire, d'écrire la mémoire (sauf pour les mémoires read-only) et d'activer la mémoire (enable). La broche enable sert principalement pour le décodage d'adresse.

La figure suivante illustre une mémoire ayant quatre mots de 3 bits…

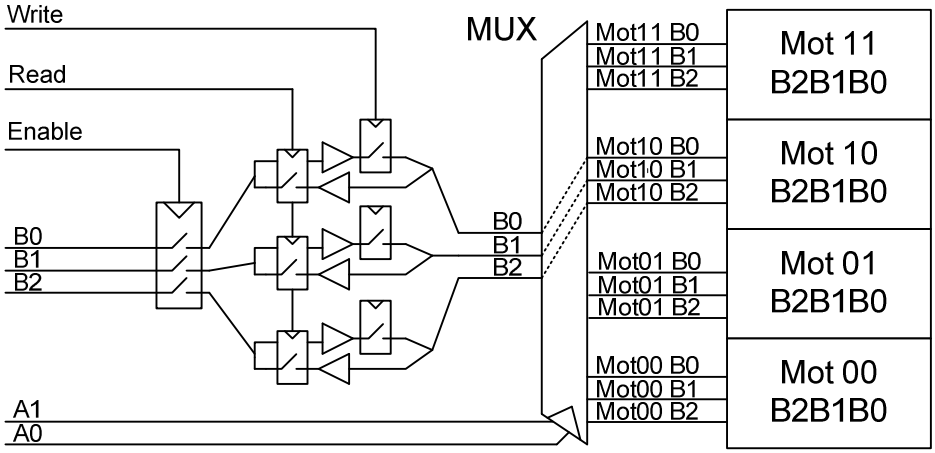

Figure 2 – Principe général de fonctionnement de la mémoire

Plus de détails sont disponibles dans la section 8, Accès à la mémoire.

2 Mémoire ROM

La Read Only Memory est habituellement écriture lors de la manufacture de la mémoire. C'est habituellement lors de sa fabrication que l'on détermine les bits qui seront lus. Il n'est pas possible de la réécrire par la suite.

Dans certains cas, les vendeurs de mémoire ROM vendent de la mémoire OTP : programmable une seule fois. Ainsi, quelqu'un intégrant la mémoire OTP dans son circuit pourra graver le contenu de la mémoire définitivement.

Il y a peu d'avantages à utiliser une mémoire OTP plutôt qu'une mémoire FLASH que l'on écrit qu'une seule fois. En cas d'erreur de programmation, la FLASH peut être récupérée… Par contre, la mémoire FLASH peut s'effacer accidentellement que ce soit en raison d'une erreur matérielle ou logicielle ou une combinaison des deux (exemple : une décharge électrostatique cause un changement aléatoire du registre PC et fait sauter dans une routine d'écriture de la FLASH).

2.1 Principe

La figure ci-dessous illustre le principe de la mémoire ROM : lors de la production de la mémoire, certains transistors de la mémoire ont été « grillés ». Ils représentent des circuits ouverts, donc des « 1 » logiques dans le circuit ci-dessous. D'autres transistors n'ont pas été grillés : ce sont des « 0 » logiques.

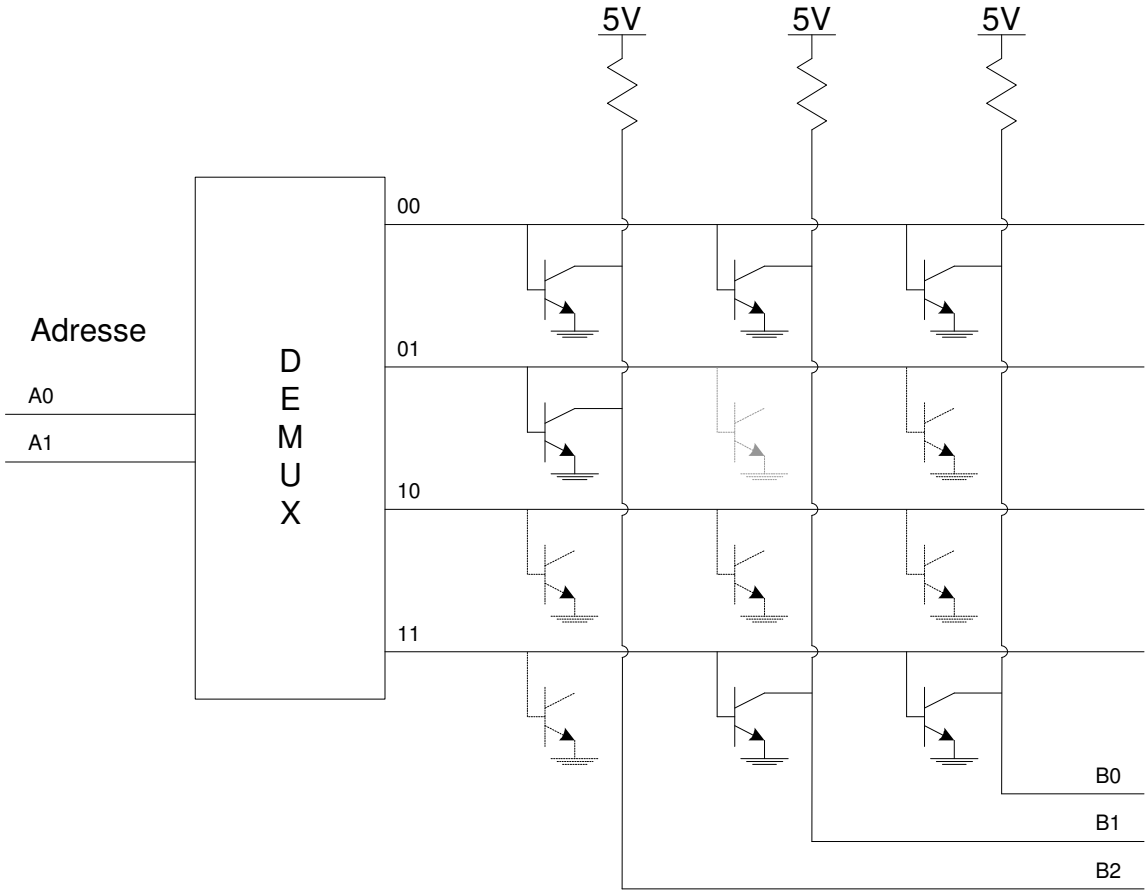

Figure 3 – Exemple de mémoire ROM OTP

Dans le circuit ci-dessus, un démultiplexeur (DEMUX) active le mot de donnée choisi en fonction de l'adresse. Par exemple, si l'adresse est ***01***, alors la ligne ***01*** sera à 5V et les autres lignes seront à 0V. Seuls les transistors de la ligne ***01*** (ceux qui n'ont pas été grillés !) imposeront des « 0 » sur les lignes de données. Les autres transistors, ayant une tension nulle sur leur base, ne conduiront pas et la tension sera 5V partout, sauf pour les bits mis à 0 par les transistors non-grillés de la ligne ***01***.

Cet exemple illustre le principe de fonctionnement de la mémoire ROM. Dans les faits, les mémoires ROM sont plus complexes (elles contiennent au moins un circuit pour mettre la mémoire en haute impédance sur le bus de données), de fabrication variées et possédant des mécanismes de gravure divers.

3 Mémoire SRAM : principe et organisation[1]

3.1 Principe

La mémoire SRAM (Static Random Access Memory) est constituée de bascules avec des transistors à effet de champs (MOS) comme illustré ci-dessous. Une bascule peut maintenir son état de sortie indépendamment de l'entrée.

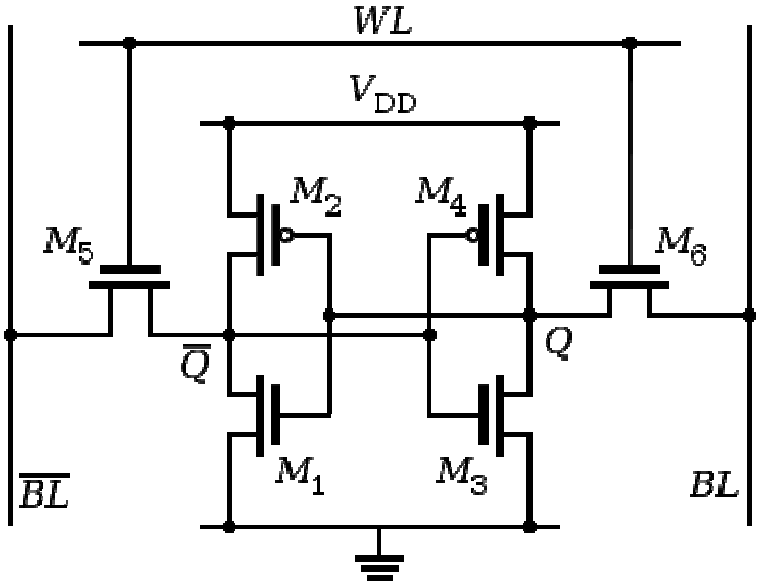

Figure 4 - Bit de mémoire SRAM

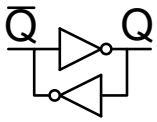

Figure 5 - Bit de mémoire SRAM, simplifié

Noter que les 2 inverseurs bouclent l'un sur l'autre (M1,M2) vs (M3,M4). M5 et M6 permettent de lire ou d'écrire (on force simplement la valeur voulue). Quand les NMOS M5 et M6 ne sont pas actifs (WL = 1 sur la Gate), Q et $\overline{Q}$ sont en état de haute impédance.

3.2 Organisation

La datasheet de la mémoire AS7C31026C sera présentée en classe.

3.3 Avantages et désavantages de la SRAM

Avantages: Ultra rapide par rapport à la DRAM (voir ci-dessous). Fiable. Simple. Idéal pour petits projets, expérimentations, mémoires caches de microprocesseur et registres.

[1] Tiré partiellement des notes de cours de M. Klein, Automne 2009

Inconvénient: Particulièrement dispendieuse. Disponible en petite quantité seulement. Consomme relativement plus que d'autres mémoires.

3.4 En savoir un peu plus sur la SRAM

Un complément à cette présentation se retrouve sur Wikipedia : les sections *Design* et *SRAM Operation* de http://en.wikipedia.org/wiki/Static_random_access_memory seront présentées en classe si le temps le permet.

4 Mémoire DRAM : principe et organisation[2]

Une cellule de mémoire DRAM (Dynamic Random Access Memory) est constituée essentiellement d'un condensateur et d'un MOSFET, ainsi qu'illustré ci-dessous.

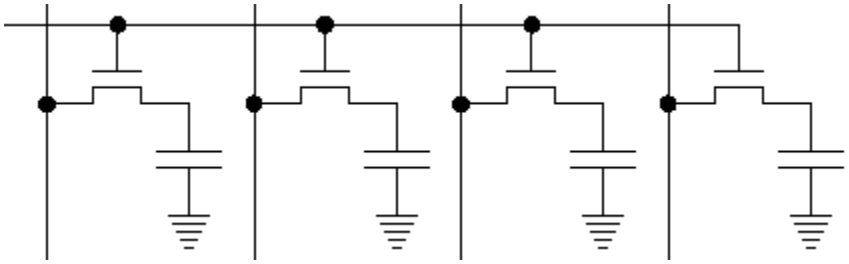

Figure 6 – Quatre bits de mémoire DRAM

Comme tous les condensateurs ont des résistances parasites et des courants de fuite, il faut recharger régulièrement les condensateurs de la mémoire DRAM afin d'éviter de perdre de l'information : c'est le rafraichissement de la DRAM.

La figure suivante montre comment les mots d'une mémoire DRAM sont organisés: toutes les gates des MOSFET sont reliées ensemble et activés simultanément lors d'un accès au mot.

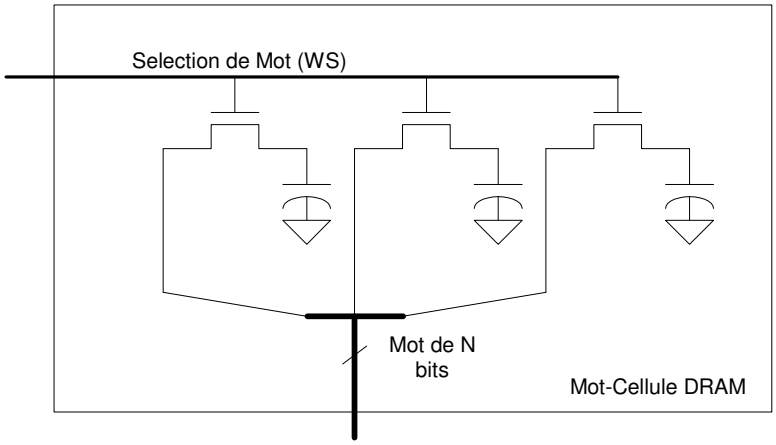

Figure 7 - Mot de mémoire DRAM

[2] Tiré partiellement des notes de cours de M. Klein, Automne 2009

Tous les mots de la mémoire DRAM sont organisés dans une matrice carrée ayant 2^N colonnes et 2^N rangées. Il y a 2^{2N} mots. Pour accéder à un mot de la mémoire, on spécifie d'abord l'adresse de la rangée du mot, puis on spécifie l'adresse colonne.

La figure suivante montre une mémoire DRAM de 16 mots (une toute petite mémoire !). Afin de lire ou d'écrire cette mémoire, il faut, d'un point de vue externe :

1) Mettre l'adresse de la rangée sur les broches d'adresse (A0 et A1) dans l'exemple.
2) Activer nRAS (Row Address Strobe, active BAS) pour saisir la rangée.
3) Mettre l'adresse de la colonne sur les broches d'adresse (A0 et A1) dans l'exemple.
4) Activer nCAS (Column Address Strobe, active BAS) pour saisir la colonne.
5) Gérer les broches de nWE et nOE pour lire ou écrire les mots, sur les broches de données (D0, D1, D2, D3), provenant de la mémoire

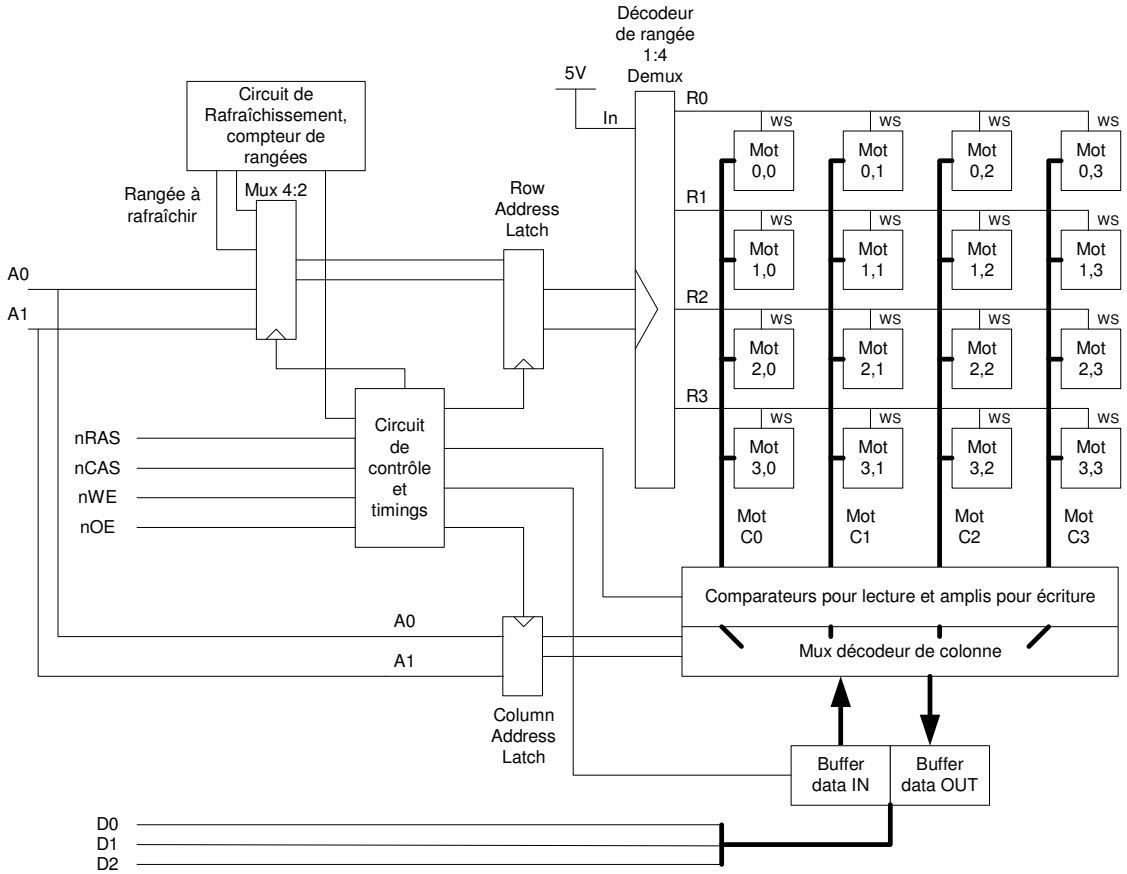

Figure 8 - Fonctionnement des mémoires DRAM

La figure suivante montre un chronographe possible d'activation de nRAS et nCAS afin d'accéder à la DRAM.

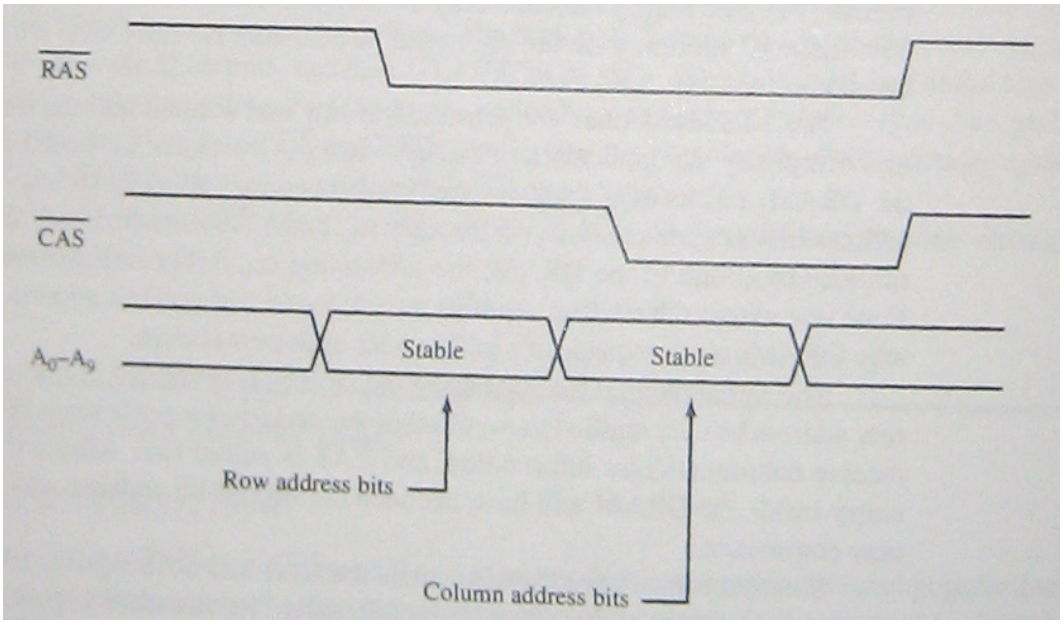

Figure 9 - Fonctionnement des mémoires DRAM

Pour une DRAM de 1 Mo (MegaOctet), il faut 2^20 adresses, ou encore 2^10 rangées et 2^10 colonnes. La DRAM requiert ainsi 10 lignes d'adresse + 2 lignes de contrôle nRAS et nCAS permettant d'entrer successivement les 10 bits hauts (nRAS = 0) puis les 10 bits bas (nRAS =0 et nCAS = 0).

4.1 Rafraîchissement de la DRAM

Le rafraichissement de la DRAM se fait généralement par un circuit de contrôle intégré à même la mémoire. Ce rafraîchissement se fait rangée par rangée : tous les mots d'une même rangée sont activés simultanément pour être rechargé.

En plus de requérir du matériel additionnel afin de gérer la rafraîchissement, celui réduit également la disponibilité de la DRAM : lorsqu'on accède aléatoirement à la DRAM, il est possible de vouloir y accéder pendant le rafraîchissement… et on doit attendre. Cela augmente le temps d'accès moyen à la mémoire.

Il existe des normes reliées aux temps de rafraichissement de la mémoire DRAM et à son opération (voir JEDEC -Joint Electron Device Engineering Council-). Selon ces normes, le rafraîchissement doit se faire à l'intérieur de 64ms ou moins.

4.2 Avantages et désavantages de la DRAM

Avantage : Très peu de composantes pour 1 bit de DRAM (un condensateur et un MOSFET) → peu dispendieux

Avantage : Requiert peu de broches pour spécifier l'adresse de mémoire visée : les lignes d'adresses sont multiplexées dans le temps. Elles indiquent la rangée et la colonne tour à tour.

Désavantage : Assez lent d'accès par rapport à la SRAM: il faut choisir la rangée, puis la colonne; il y a plusieurs transistors entre le bit accédé et la broche de donnée (le temps de propagation du signal est assez important quand on parle de nanosecondes!); et, finalement, il faut rafraichir la DRAM.

4.3 En savoir un peu plus sur la DRAM

Un complément à ce document se retrouve sur Wikipedia : vous êtes invités à lire http://en.wikipedia.org/wiki/Dynamic_random_access_memory pour en savoir plus.

Si le temps le permet, la DRAM MT4LC4M16R6 sera présentée en classe.

5 SDRAM et autres mémoires DRAM

La SDRAM est de la DRAM synchrone, c'est-à-dire régie par un signal d'horloge. La mémoire SDRAM est agencée en banques de mémoire DRAM indépendantes. Un contrôleur de mémoire interface entre les banques et le microprocesseur qui peut faire différentes requêtes. Le microcontrôleur (à travers un module d'interface) peut, par exemple, demander des bursts de lectures (plusieurs lectures en rafale) de plusieurs banques différentes sans attendre nécessairement que les lectures d'une banque soient terminées. Le microcontrôleur peut aussi demander (toujours par le biais d'un module d'interface), un rafraichissement de la DRAM d'une banque…

Les signaux de contrôle de la SDRAM ont des noms similaires à ceux de la DRAM, mais n'opèrent pas de la même façon : l'ensemble des signaux définit une commande à être exécutée par la mémoire SDRAM.

La datasheet de la mémoire MT48LC128M4A2 sera présentée brièvement en classe afin d'illustrer les propos ci-dessous.

Si le temps le permet, plus d'informations seront données sur la mémoire SDRAM (voir http://en.wikipedia.org/wiki/Synchronous_dynamic_random_access_memory).

5.1 Autres mémoires DRAM

On retrouve la mémoire DRAM sous plusieurs formes dans l'industrie. Cette mémoire est peu dispendieuse et beaucoup de développement a été fait avec cette technologie. C'est la mémoire vive de vos ordinateurs. Voici quelques exemples de mémoires DRAM qu'on retrouve dans le marché :

- PSRAM : Pseudo-static RAM. Il s'agit d'une mémoire DRAM qui inclut un circuit de rafraîchissement et un système de gestion des rangées/colonne différent. La PSRAM offre la même interface que la mémoire SRAM.
- EDO : Prédécesseur de la mémoire DRAM. Dinosaure maintenant éteint.
- VRAM : Video RAM. La mémoire SDRAM se retrouve aussi dans les cartes graphiques !
- DDRx et XDR : Mémoire SDRAM cadencées de plus en plus vite : sur les deux fronts montant et descendant du signal d'horloge, avec un bus plus large, avec une horloge plus rapide..,

6 Mémoires Non-Volatiles (EEPROM, FLASH et FRAM)

Il existe plusieurs mémoires non-volatiles sur le marché. En dehors de la mémoire sur bande magnétique (disque dur), les mémoires non-volatiles les plus fréquentes sont les mémoires FLASH, EEPROM et FRAM.

Les caractéristiques les plus importantes des mémoires non-volatiles modernes sont : le temps de lecture (le temps pris pour lire une adresse de mémoire), le temps d'écriture (le temps pris pour écrire une adresse de mémoire), la taille des pages (les octets sont regroupées en pages lorsque les données sont effacées), le nombre de fois (cycles) où l'on peut écrire la mémoire et, évidemment, le coût.

La mémoire EEPROM est la plus vieille mémoire parmi les trois mémoires ci-dessus. Il s'agit d'une mémoire assez dispendieuse (par rapport à la FLASH), qui s'efface électriquement. Le temps de lecture de la mémoire EEPROM est très rapide (quelques ns), mais le temps d'écriture est souvent très long (de l'ordre de 5ms). Il est possible d'écrire un seul octet à la fois, même si les données sont regroupées en pages qui ont habituellement autour de 128 octets. Lorsque les données à écrire sont sur deux pages différentes, il faut faire deux opérations d'écriture différentes. L'EEPROM supporte habituellement entre 100 000 et 1000 000 cycles d'écriture.

La mémoire FLASH est plus récente que la mémoire EEPROM. Elle est moins dispendieuse et elle a des temps de lecture équivalents à l'EEPROM. Le principal désavantage de la FLASH est l'écriture : il n'est possible d'écrire la FLASH que par pages (habituellement 512 bytes) et l'écriture de la FLASH est très longue. La mémoire FLASH se retrouve comme mémoire d'instructions (les instructions ne sont pas écrites souvent !!!) dans la plupart des microcontrôleurs modernes. La FLASH supporte habituellement entre 100 000 et 1000 000 cycles d'écriture.

La mémoire FRAM est une mémoire d'avenir. Il s'agit d'une technologie récente et elle est encore très couteuse (par rapport à l'EEPROM ou la FLASH). La mémoire FRAM est conçue comme la mémoire DRAM, mais elle est non-volatile (et statique). Les temps de lecture et d'écriture de la FRAM sont similaires à ceux de la DRAM, ce qui fait que l'écriture de la FRAM est beaucoup plus rapide que celle de la FLASH ou celle de l'EEPROM. L'écriture se fait aussi par octet, avec un nombre de cycles d'écriture très petit. Il s'agit d'une mémoire d'avenir, si son prix diminue…

6.1 Un Bit de FLASH

Les bits de FLASH sont créées à partir de FGMOS : Une grille flottante (isolée électriquement) est chargée ou déchargée afin d'indiquer si le transistor contient un « 0 » ou un « 1 »

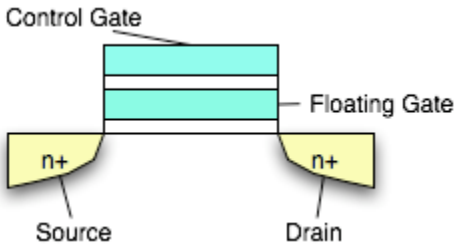

Figure 10 – FGMOS (http://en.wikipedia.org/wiki/File:Floating_gate_transistor.png)

Afin d'évaluer la valeur du bit contenu dans le FGMOS, une tension prédeterminée est appliquée sur la grille de contrôle (Control Gate). En fonction de la charge présente sur la grille flottante, la tension de contrôle créera ou ne créera pas de canal d'électrons permettant la circulation de courant entre la Source et le Drain.

Deux techniques sont utilisées afin de charger ou décharger la grille flottante : « Fowler-Nordheim tunneling » et « hot carrier injection ». Bien que la description de ces méthodes ne soient pas un sujet du cours de SMI, il faut retenir que l'effacement et l'écriture de FLASH se fait avec des tensions spéciales (exemple : 12V) et que cela peut implique de forts courants.

6.2 NOR FLASH et NAND FLASH

Il y a deux types de FLASH retrouvés communément dans l'industrie : la FLASH NAND et la FLASH NOR. La NOR permet des accès en lecture aléatoire, c'est-à-dire n'importe où, rapidement. La NAND ne se lit que par page et la lecture est légèrement plus lente. Cependant, la NAND est moins dispendieuse que la NOR.

6.2.1 NOR FLASH

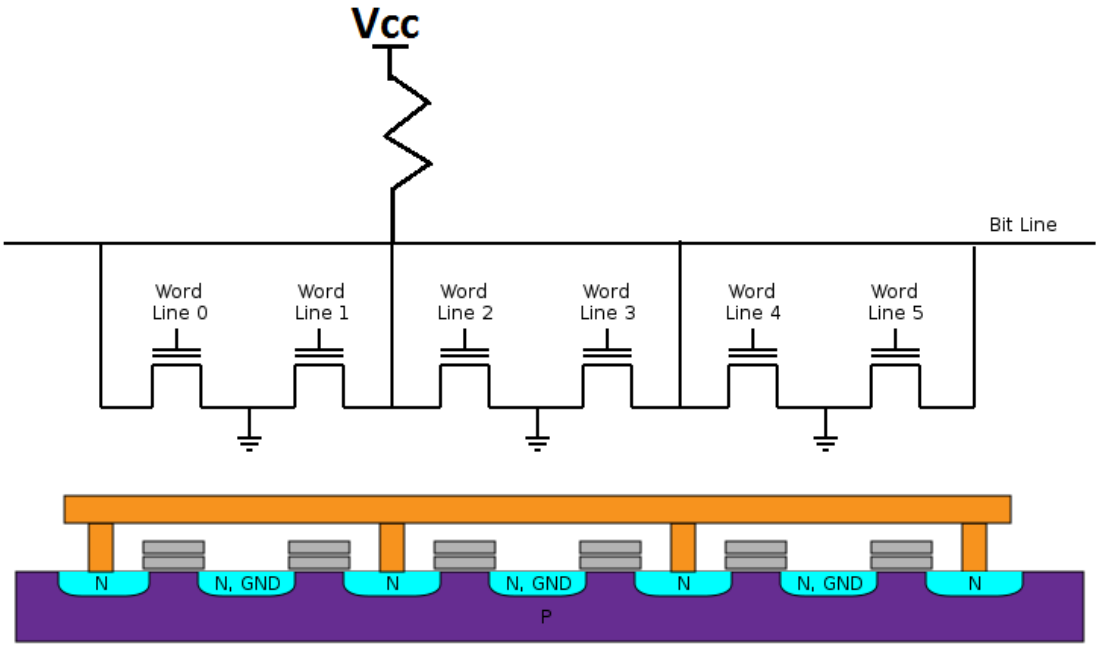

Figure 11 – NOR FLASH (http://en.wikipedia.org/wiki/File:NOR_flash_layout.svg)

Comme illustré ci-dessus, tous les bits de même poids des différents mots de la NOR sont connectés en parallèle.

Lorsqu'un mot est lu (Mot 1 par exemple), la ligne du mot est activée (on met une tension seuil sur la ligne du Mot) et toutes les autres lignes sont désactivées (0V par rapport au GND). En fonction de la charge de la grille flottante associée au bit lu, le FGMOS conduira ou ne conduira pas, forçant la ligne de bit à 0V ou la laissant à Vcc.

Il s'agit d'une NOR FLASH parce que toutes les entrées doivent être à 0 (ou tous les FGMOS ne doivent pas conduire) pour que la sortie soit à 1.

6.2.2 NAND FLASH

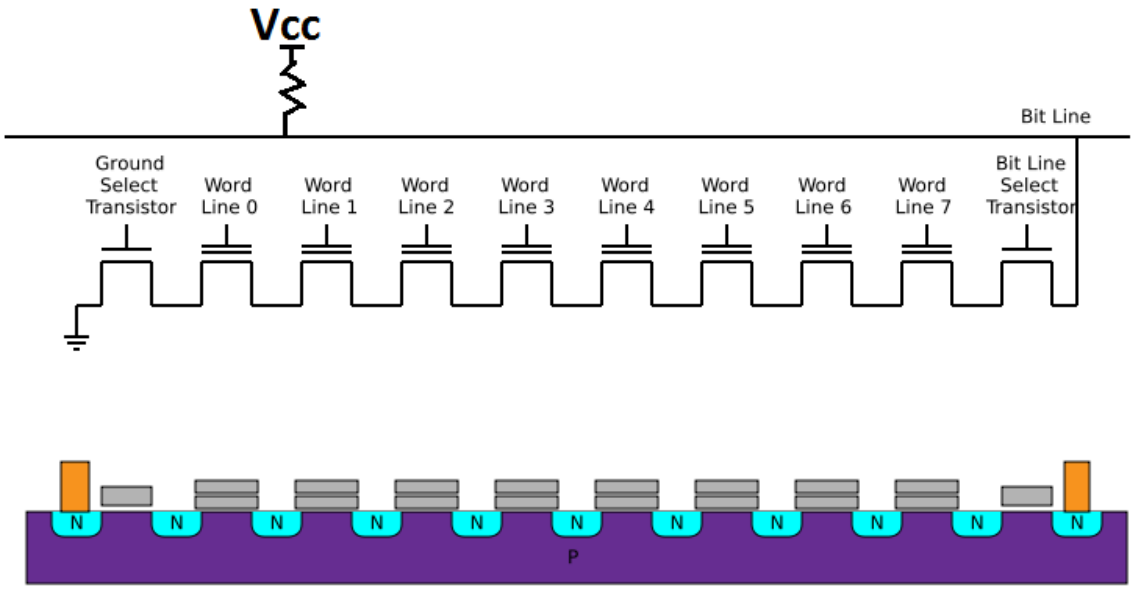

Figure 12 – NAND FLASH (http://en.wikipedia.org/wiki/File:Nand_flash_structure.svg)

Les FGMOS associés à un bit de poids donné pour différents mots sont tous en série avec la NAND FLASH.

Lorsqu'un bit d'un mot est lu, la tension de seuil de conduction du FGMOS est appliquée sur le mot lu. Par ailleurs, une tension supérieure à la tension de seuil est appliquée sur les autres mots de telle sorte que tous les FGMOS conduisent (peu importe la charge de leur grille flottante), sauf, peut-être celui du mot choisi. Ce dernier conduira ou ne conduira pas selon la charge de la grille flottante.

Il s'agit de NAND FLASH parce que la ligne de bit sera « 0 » uniquement lorsque tous les mots de contrôles seront « 1 », c'est-à-dire uniquement lorsque tous les FGMOS conduiront.

6.2.3 Implémentation de FLASH

Si le temps le permet, plus d'information sur la FLASH sera présentée à partir du site http://en.wikipedia.org/wiki/Flash_memory ou d'une datasheet de mémoire FLASH.

7 Décodage d'adresse

« Lorsque le CPU veut parler à un périphérique (ram, port série, un autre CPU etc.) il doit d'abord l'adresser, c'est-à-dire donner mettre l'adresse de ce périphérique sur le bus d'adresse.

C'est alors à tous les périphériques sur le BUS d'adresse de lire l'adresse. Chaque périphérique doit alors comprendre si l'adresse en cours s'adresse à lui ou non. Ce processus est le décodage d'adresse.

Le décodage d'adresse fait partie de la **glue logique** entre le BUS d'adresse et le périphérique. Le décodeur n'est presque jamais inclus avec ou dans le périphérique car sa forme dépend trop de l'architecture générale du système microprocesseur.

Un bloc de décodage d'adresse gère en général plusieurs périphériques ou mémoires.

La sortie du bloc de décodage est habituellement N broches, N étant le nombre de périphériques et de mémoires, de 1 à N.

Si l'adresse sur le BUS indique le périphérique #3, alors la glue logique devait activer sa broche #3 et désactiver les autres, la broche #3 étant reliée à la broche d'activation du périphérique #3.

En effet, les broches de sortie du décodeur sont connectées au ENABLE (ou équivalent: on voit souvent des noms comme E^, EN^, SEL^, CS^ ..) des périphériques sur le BUS. Ainsi le décodeur active le périphérique adressé et désactive (Hi-Z) tous les autres. »[3]

Habituellement, une plage d'adresse est attribuée à chaque périphérique. Supposons, par exemple, un système microprocesseur avec 20 bits d'adresse, avec une mémoire ROM de 64k*8 octets, une mémoire RAM de 8k*8 octets et des périphériques ayant besoin de 1k*8 ports. Les adresses de 0x00000 à 0x0FFFF pourront être attribuée à la ROM (0x0FFFF – 0x0000 + 1 = 0x10000 = 64k), les adresses de 0x20000 à 0x21FFF pourront être attribuées à la RAM et les adresses 0x40000 à 0x403FF pourront être attribuée aux périphériques.

Décoder l'adresse d'une mémoire consiste donc à activer la mémoire de telle sorte que cette mémoire soit connectée au bus de donnée lorsque le microprocesseur veut accéder à cette mémoire (en fonction de l'adresse de l'instruction ou de la donnée lue/écrite).

Pour décoder l'adresse de la mémoire, le bus d'adresse est utilisé en deux parties : une première partie désigne la mémoire et l'autre partie désigne la position à l'intérieur de la plage de mémoire. Les bits désignant la position à l'intérieur de la mémoire ne sont pas utilisés dans le décodage d'adresse.

Lorsque plusieurs adresses désignent la même plage de mémoire, on parle de décodage partiel. Dans ce cas, certaines lignes d'adresse ne servent pas à désigner une position dans la plage de mémoire, ni à décoder l'adresse.

Quelques exemples de décodage d'adresse seront présentés en classe (voir Memoires_Materiel_DecodeAdr.ppt)

> *Dans le texte ci-dessus, le terme "périphérique" inclut la mémoire et les périphériques tel que défini dans les autres sections des notes de cours. Il s'agit d'un abus de langage acceptable dans le but d'alléger le texte. "Périphérique" est plus*

[3] Tiré partiellement des notes de cours de M. Klein, Automne 2009

généralement défini comme un appareil connecté à un ordinateur et le terme exclut habituellement la mémoire, mais inclut les disques durs.

8 Accès à la mémoire

Lors d'une lecture de la mémoire les étapes suivantes ont toujours lieu :

R1a) Mettre l'adresse de la donnée à lire sur le bus d'adresse
R1b) Sélectionner la mémoire (se fait habituellement avec le décodage d'adresse)
R1c) Active le signal de lecture au besoin.
R2) Attendre que la mémoire mette la donnée sur le bus de donnée
R3) Lire la donnée sur le bus de donnée

Lors d'une écriture de la mémoire les étapes suivantes ont toujours lieu :

E1a) Mettre l'adresse de la donnée à lire ou écrire sur le bus d'adresse
E1b) Mettre les données à écrire dans la mémoire sur le bus de donnée
E2a) Activer la mémoire (se fait habituellement avec le décodage d'adresse)
E2b) Activer le signal d'écriture au besoin
E3) Attendre que la mémoire ait lu les données.

Plusieurs étapes peuvent s'insérer entre chacune de ses étapes, en fonction de l'architecture du bus. Plusieurs broches additionnelles peuvent aussi gérer le flot de données entre la mémoire et le microprocesseur. Voici quelques exemples :

- Il est possible d'activer une broche Address Latch Enable si les broches de données et d'adresse sont multiplexées dans temps. Cette broche détermine le rôle des broches AD. On retrouve aussi parfois Address Strobe qui indique qu'une adresse valide se retrouve sur le bus.
- Des broches permettent aussi parfois de choisir les octets à lire/écrire lorsque le mot de mémoire est de plusieurs octets.
- On retrouve également, dans certains systèmes, des broches qui permettent de choisir entre la mémoire et les entrées/sorties.
- Les lignes de contrôle RAS et CAS vues avec la DRAM sont également des broches additionnelles qui gèrent la mémoire.
- Enfin, il peut y avoir une ligne de contrôle, gérée par la mémoire qui indique que les données sont prêtes à être lues par le microprocesseur…
- Etc.

Bref, il y a plusieurs possibilités et plusieurs séquences possibles d'activation et désactivation des broches afin d'accéder à la mémoire.

8.1 Chronographe

Un chronographe est une figure dont l'axe des « x » est le temps et dont l'axe des « y » comporte plusieurs signaux. Chaque signal sur l'axe des « y » varie dans le temps et une séquence d'évènement est illustrée sur les chronographes.

Le chronographe de la mémoire AS7C31026C sera présenté en classe.

9 Architectures de système mémoire-microprocesseur

Lorsqu'on conçoit un système mémoire-microcontrôleur, plusieurs options sont possibles. Trois options communes sont présentées ici : Mémoire interne/externe, architecture Harvard, et mémoire série/parallèle.

9.1 Mémoire interne et externe

La mémoire utilisée pour les instructions ou les données peut être interne ou externe. Habituellement, les mémoires internes sont plus rapides, moins couteuses et elles ne prennent presque pas d'espace, mais elles sont de tailles fixes.

9.2 Architecture Harvard versus Von Neumann

Dans l'architecture Von Neumann, un seul bus permet d'accéder à la mémoire de donnée, à la mémoire d'instructions et aux entrées sorties. Dans l'architecture Harvard, un bus permet d'accéder à la mémoire d'instructions alors qu'un autre bus permet d'accéder à la mémoire de données et aux entrées sorties.

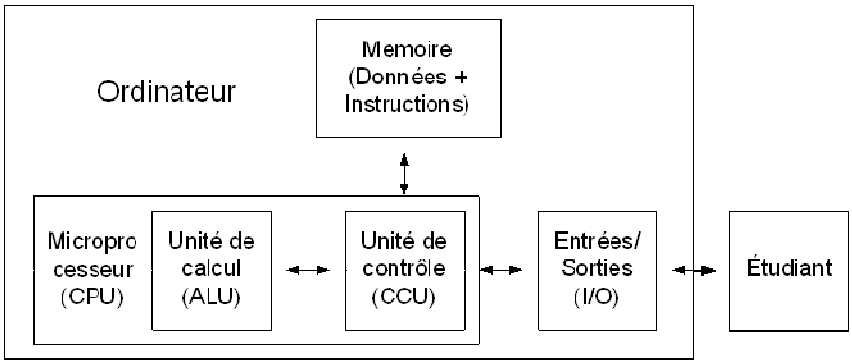

Figure 13 – Architecture Von Neumann

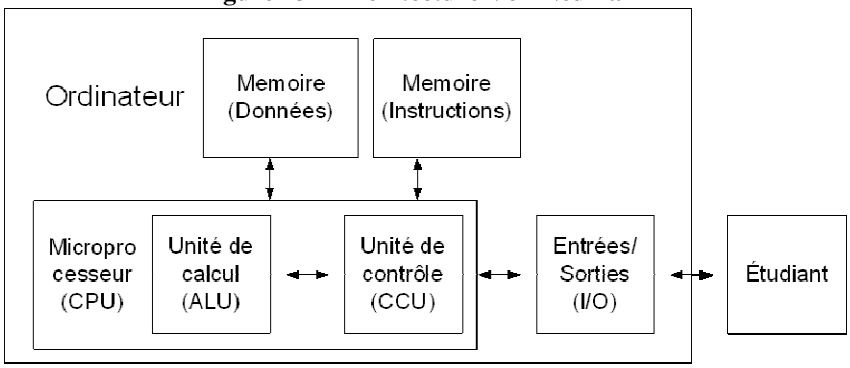

Figure 14 – Architecture Harvard

9.3 Mémoires séries et mémoires parallèles

Les mémoires peuvent être accédées par un bus série ou un bus parallèle.

Un bus série est un bus ou les bits d'un mot sont transmis consécutivement. Un bus parallèle est un bus sur lequel les bits d'un mot sont transmis simultanément.

Le chronographe de la mémoire 25LC512 sera présenté en classe.

10 DMA (Direct Memory Access)

10.1 Transfert de données par DMA (Rappel)

L'accès direct à la mémoire est une technique qui permet de transférer des données directement entre un périphérique et la mémoire (ou l'inverse) sans l'intervention du microprocesseur. Le DMA requiert un circuit spécial, le contrôleur de DMA, qui s'occupe d'effectuer les transferts entre la mémoire et un périphérique.

Le contrôleur de DMA exécute généralement un transfert de données plus vite que le microprocesseur.

En effet, pour effectuer un transfert de données, le microprocesseur lit une instruction lui disant de lire un mot de données. Il exécute cette instruction. Puis, il lit une instruction lui disant d'écrire le mot de données lues précédemment. Ensuite, le microprocesseur lit une ou des instructions afin de savoir si le transfert est terminé d'incrémenter les adresses visées. Il exécute ces instructions et continue le transfert s'il y a lieu.

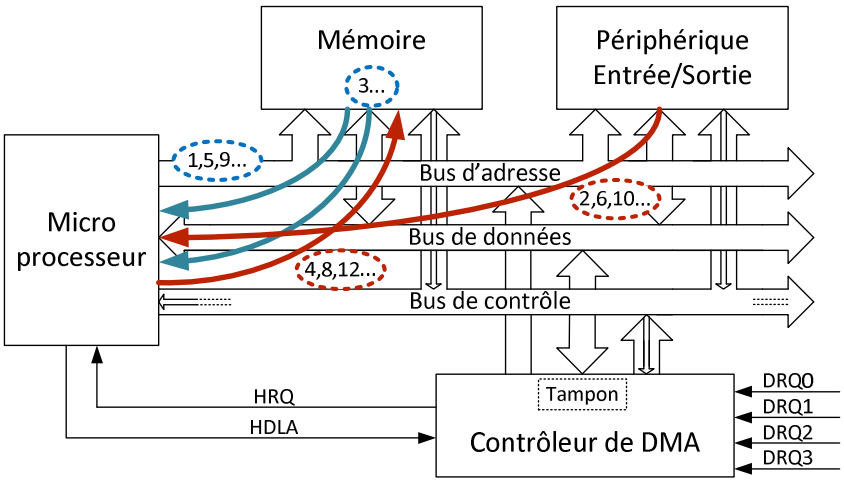

Figure 15 – Transfert de données d'un périphérique à la mémoire par le microprocesseur

Dans la figure ci-dessus, les étapes impaires du transfert (1, 3, 5…), illustrées en bleu, sont des lectures d'instruction. Les étapes paires, illustrées en marron sont des transferts de données. Lors des étapes 2, 6, 10…, la donnée du périphérique est mise dans un registre tout usage du microprocesseur. Lors des étapes 4, 8, 12, … la donnée est écrite dans la mémoire à partir du registre du microcontrôleur.

Le contrôleur de DMA a une approche plus simple. Lorsqu'un transfert est requis, le contrôleur de DMA négocie l'accès au bus de la mémoire avec le microprocesseur. Quand le microprocesseur cède le bus au contrôleur de DMA, le contrôleur DMA prend alors le contrôle du bus d'adresse et de contrôle. Il met l'adresse de la source des données sur le bus d'adresse et lit la donnée de la source. Il incrémente l'adresse de la source des

données. Puis, le contrôleur de DMA met l'adresse de la destination sur le bus d'adresse et écrit la donnée dans la destination. Le contrôleur de DMA incrémente l'adresse de destination. Enfin, il vérifie si le transfert est terminé en fonction des adresses ou du nombre de mots de données transférés.

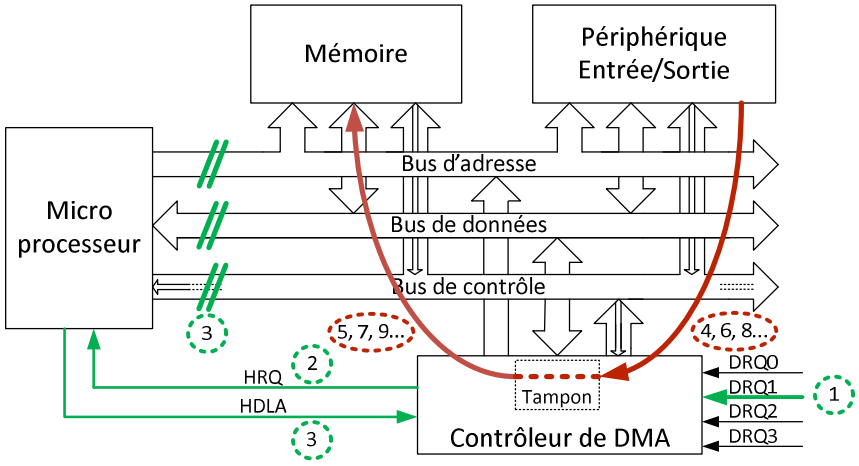

Figure 16 – Illustration simplifiée du transfert par DMA

La figure ci-dessus illustre un transfert par DMA, de façon très simplifiée. Les étapes du transfert sont les suivantes :

0) Des instructions de la mémoire disent au microprocesseur d'écrire des registres du contrôleur de DMA afin de configurer et préparer les transferts. Des instructions écrivent aussi certains registres du microprocesseur également afin de configurer les transferts par DMA. Enfin, d'autres instructions exécutées par le microprocesseur écrivent des registres des périphériques visés par la DMA, toujours afin de configurer le transfert par DMA.
1) À un moment donné, un périphérique ou le microprocesseur réclame un transfert par DMA. Il active une broche reliée au contrôleur de DMA (DRQ1 dans l'exemple de la figure)
2) Le contrôleur de DMA négocie l'accès au bus avec le microprocesseur (Dans l'exemple, le contrôleur de DMA signale HRQ. Il obtient le bus si/lorsque le microprocesseur signale HLDA).
3) Le contrôleur de DMA initie le transfert de données : il établit les adresses de base de la source et de la destination.
4) Le contrôleur de DMA exécute le transfert de données : il incrémente les adresses visées et gère le bus de contrôle jusqu'à ce que le transfert soit terminé.

Dans la figure, les données du périphérique sont transférées une-à-une vers la mémoire, en passant par un tampon. Ce tampon est nécessaire parce qu'il faut des adresses différentes pour la destination et pour la source (le bus d'adresse ne peut contenir deux adresses différentes simultanément!). Par ailleurs, la mémoire et le périphérique ne vont pas à la même vitesse : il faut une mémoire tampon pour entreposer les données.

Enfin, la figure illustre ce qu'est le DMA, mais elle ne représente pas la plupart des systèmes modernes avec DMA. La section suivante décrit pourquoi.

10.2 Activité du microprocesseur pendant le DMA

Dans la section précédente, le DMA est utilisé dans avec un ordinateur ayant une architecture Von Neumann et le microprocesseur est inactif pendant tout le transfert par DMA. Cette façon de faire était commune dans les années 80 car même si le microprocesseur est inactif (HRQ = Hold ReQuest), le DMA reste efficace par rapport à un transfert par des instructions. Toutefois, dans la plupart des systèmes avec DMA modernes, le transfert par DMA s'effectue en parallèle avec l'exécution d'instructions par le microprocesseur. Plusieurs stratégies sont possibles :

- Dans les architectures Harvard, le DMA transfère des données des périphériques vers la mémoire de données et vice-versa. Pendant le transfert par DMA, le microprocesseur a toujours le contrôle du bus d'instruction et il peut continuer d'opérer, tant que les instructions ne font pas d'accès aux données, c'est-à-dire à la mémoire.

Un exemple typique d'architecture Harvard où le DMA est utilisé est le traitement de signal. Les DSPs (Digital Signal Processors) acquièrent des données à partir des périphériques tels que les ADCs. Les données sont ensuite transmises en mémoire pour y être traitées subséquemment. En utilisant le DMA, il est possible de transférer les données nouvellement acquises par un périphérique tout en traitant d'anciennes données.

- Dans plusieurs systèmes microprocesseurs modernes, le microprocesseur peut accéder à la mémoire, même lorsqu'un transfert DMA a lieu, tant que le microprocesseur n'accède pas à une région précise de la mémoire, celle où le transfert par DMA à lieu. Plusieurs techniques permettent cela: des mémoires ayant plusieurs canaux d'accès, des caches ou tampons spécialisées, des matrices de bus et plus.

- Dans d'autres systèmes, le temps d'accès au bus de la mémoire est fragmenté en petits intervalles : dès que le microprocesseur n'exécute pas une instruction LOAD/STORE, le contrôleur de DMA effectue une partie du transfert…

Bref, plusieurs stratégies sont possibles afin de permettre l'opération du microprocesseur pendant un transfert par DMA.

10.3 Configuration du DMA et types de transfert

Le cours *SMI_C13_Programmation_Peripheriques* présente des détails additionnels sur la configuration du DMA et les types de transfert par DMA.

10.4 Applications du DMA

Le DMA a plusieurs applications. Il est particulièrement utile lorsqu'un transfert de données rapide est requis (plus rapide qu'avec des instructions) ou lorsqu'on veut libérer le microprocesseur de cette tâche afin qu'il exécute autre chose.

Un premier exemple illustration cette idée est lorsque des données sont transférées du disque dur vers la mémoire : on veut que le transfert soit rapide afin de diminuer le temps d'accès au disque dur et le DMA est utilisé dans ce cas.

Un second exemple illustrant l'idée ci-dessus est lors de communication radio : le DMA prend les échantillons de signal reçu et les place en mémoire tandis que le microprocesseur traite les données en mémoire afin de décoder les messages reçus.

11 Memory Management Unit

Plusieurs opérations peuvent être nécessaires avant d'accéder à la mémoire : traduire l'adresse virtuelle du programme en adresse physique, vérifier si l'information voulue est dans les caches, vérifier si le processus a les permissions nécessaires pour accéder à l'adresse de mémoire, vérifier/négocier l'accès au bus de mémoire… Habituellement, ces tâches sont gérées par le MMU (Memory Management Unit) qui est un circuit logique habituellement intégré au même die que le microprocesseur.

Si le temps le permet, plus de détails seront donnés en classe sur chacun des éléments suivants.

11.1 Adresse virtuelle et translation d'adresse

Dans un système où les programmes sont (re)localisés en mémoire par le système d'exploitation, toutes les instructions/données des programmes ont deux adresses : une adresse virtuelle (celle relative au début du programme) et une adresse physique, c'est-à-dire l'adresse où se retrouve l'instruction/donnée à l'intérieur de la mémoire.

Comme la translation d'adresse (le passage de l'adresse virtuelle à l'adresse physique) est une tâche parfois compliquée et comme cette tâche doit être faite pour toutes les instructions/données, la translation d'adresse est effectuée par du matériel spécialisé à l'intérieur du MMU.

11.2 Protection de la mémoire

La mémoire est souvent découpée en segments/pages par le système d'exploitation et le microprocesseur. Ces segments sont souvent protégés : certains segments de mémoire ne sont accessibles que par le système d'exploitation alors que d'autres segments, réservés aux programmes de l'usager, sont accessibles par tous.

La protection de la mémoire est nécessaire pour éviter que les programmes ne viennent perturber le bon fonctionnement du système d'exploitation et pour éviter que les programmes interagissent involontairement entre eux.

Comme protéger la mémoire est une tâche qui demande un certain temps et demande l'accès à l'information de protection, cette tâche est aussi effectuée par du matériel spécialisé à l'intérieur du MMU.

11.3 Arbitration de bus

Dans les systèmes multiprocesseurs, des données ou des instructions peuvent se retrouver dans des mémoires distribuées à travers tout le système. Un ou plusieurs bus peuvent servir à communiquer avec ces mémoires et le MMU en négocie l'accès. Il peut également y avoir un nombre restreint de bus pour accéder à une mémoire unique et partagée. Dans ce cas, le MMU doit aussi négocier l'accès à cette mémoire.

Enfin, le MMU peut avoir à négocier l'accès avec au bus système lorsque d'autres contrôleurs de bus, comme le DMA, peuvent en prendre le contrôle. Le contrôleur de DMA et le MMU sont toujours étroitement reliés dans un système microprocesseur.

11.4 Contrôle de caches

Les caches contiennent les données/instructions les plus susceptibles d'être accéder prochainement afin de réduire les temps d'accès à la mémoire. Les caches contiennent des copies des données en mémoires, et seulement une fraction des données de la mémoire (les caches sont plus rapides que la mémoire, mais plus couteuses et donc plus petite!). Dans un système de mémoire avec des caches, il faut trouver les données dans les caches si elles y sont. Il faut aussi mettre à jour les données écrites dans les caches et gérer le transfert d'information entre les caches et la mémoire. Traditionnellement, ces rôles sont réservés au MMU.