Chapitre 1. Approche des circuits programmables (1 semaine)
Architecture de base, Modèle de Von Neumann, l'unité centrale, la mémoire principale, les interfaces d'entrées/sorties, les bus, décodage d'adresses
Chapitre 2. Architecture d'un microprocesseur 16 bits (MC68000) (5 semaines)
Architecture interne, Brochage, Registres spéciaux, Modes d'adressages, Jeux d'instructions, Différentes architectures : RISC, CISC, Harvard

Chapitre 1. Approche des circuits programmables

Avant propos:

Pour entamer le cours sur les microprocesseurs vous devez faire une révision sur la matière LCS (logique combinatoire et séquentielle) et la matière principe de programmation en langage C. vous devez connaitre ce que c'est un circuit combinatoire tels que le décodeur le multiplexeur, l'additionneur etc. et un circuit séquentiel tels que les bascules, les compteurs et les registres car le microprocesseur est un circuit qui englobe les deux. Vous devez aussi maitriser la conversion du décimal au binaire et à l'hexadécimal et vice versa. Quant à la programmation en langage C vous devez connaitre la notion d'algorithmique et le processus de développement d'un programme en langage évolué tel que l'éditeur de texte, la compilation, l'édition de liens le chargement et l'exécution.

Rappel : Circuits séquentiels

Un circuit combinatoire, formé par des portes logiques, est simplement une fonction qui calcule des valeurs de sortie en fonction des valeurs d'entrée. Il n'y a aucune notion de temps dans les circuits combinatoires ni de contreréaction (feedback : sortie réinjectée a l'entrée). .
Un circuit séquentiel est un circuit combinatoire auquel on a ajouté la notion de temps avec un signal d'horloge. Un signal d'horloge est un signal carré périodique qui peut être produit par un quartz.

Signal d'horloge

Les circuits programmables

les fabricants de circuits intégrés ont donné naissance aux Circuits Logique Programmable ou encore P.L.D.(Programmable Logic Device). On distingue deux grandes catégories :
les circuits logiques reconfigurables (Il n'y a pas de programmation en logiciel dans un certain sens) et **les circuits programmables** dans le sens d'utiliser l'exécution d'un programme pour déterminer la réalisation d'un traitement ciblé.

1-Circuit logique reconfigurable
C'est un circuit intégré qui fonctionne de manière logique et qui permet sa reprogrammation à volonté même après sa fabrication. On utilise plutôt le terme « reconfiguration » plutôt que « reprogrammation » (car ses connexions ou ses composants, sont à base des portes logiques et des bascules). Le terme programmation est utilisé fréquemment, pour personnaliser les réseaux logiques reconfigurables exemple PAL (Programmable Array Logic) : réseaux logiques programmables
- La programmation se fait par destruction de fusibles.
-Aucun fusible n'est grillé à l'achat de la PAL.
A partir de cette structure de base il va être possible de réaliser de nombreuses fonctions logiques. La programmation va constituer à détruire les fusibles pour obtenir les fonctions désirées, en sachant que lors de l'achat d'un P.A.L.tous les fusibles sont vierges ou pas détruits.

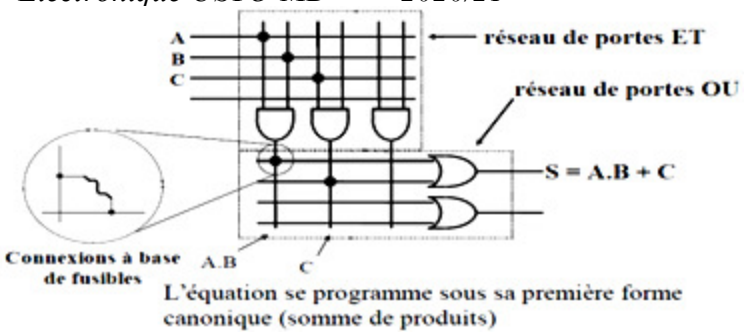
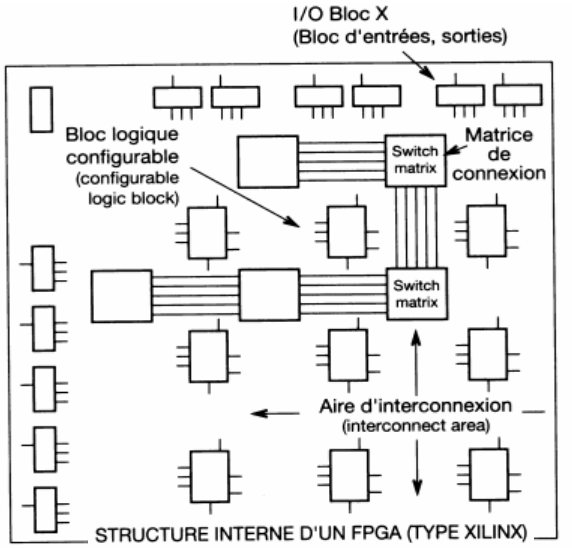


Fig1. Réseaux logiques programmables (programmation de la fonction S=AB+C

L'autre exemple le plus utilisé est le FPGA : composés de plusieurs milliers voire millions de portes logiques et de bascules. Les deux plus grands constructeurs de FPGA sont XILINX et ALTERA. Ils sont composés de blocs logiques élémentaires qui peuvent être interconnectés.



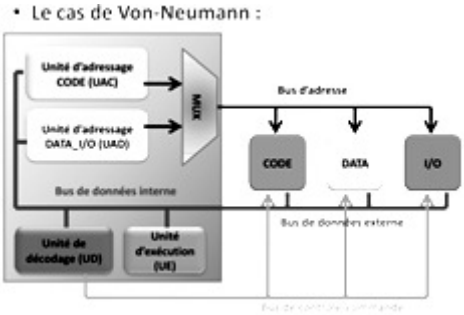
La description du fonctionnement des circuits peut se faire soit:
- Par un schéma à base de fonctions logique élémentaires (Portes ET,OU,NON, … bascules, compteurs, registres à décalages).
- En utilisant un langage de description comportementale H.D.L. (HardwareDescription Language). VHDL (Very high speed Hardware Description Language) et VERILOG sont en général utilisés pour des circuits complexes.

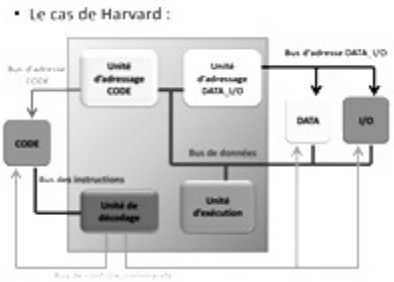
2-Circuit de traitement programmable : Le processeur

Cette appellation est tirée du mot anglais to process qui veut dire traiter
Compte tenu de sa taille très petite comparativement à la fonction de traitement qu'il effectue et en prenant en considération le grand avancement technologique dans la fabrication des circuits intégrés qui est passée de l'intégration à petite échelle (SSI) à l'intégration à très grande échelle (VLSI), ce type de circuit est appelé **microprocesseur**. C'est un circuit intégré très petit qui fait un traitement très varié et très sophistiqué comprenant un très grand nombre de transistors qui sont à la base de la construction de n'importe quelle fonction logique (Rappelez vous avec la porte NAND on construit en logique tout ce que vous pouvez imaginer (LCS)).
Dans la suite on utilise processeur ou microprocesseur pour designer la même chose et on sous entend l'unité centrale ou CPU (central processing unit).

2 Architecture de base

le processeur ne fonctionne jamais tout seul car sa fonction est le traitement, ce traitement est exprimé sous forme d'un programme, ce programme doit être mémorisé ou stocké dans une mémoire, donc la mémoire est un composant nécessaire pour le fonctionnement d'un processeur. Pour interagir avec le processeur on a aussi besoin des interfaces (intermédiaires entre l'humain et le circuit) ces unités on les appelle les périphériques ils sont a proximité (prés et à la périphérie) du processeur. Les trois composants sont reliés entre eux par des bus (ensemble de conducteurs) on distingue trois bus standards transportant des informations différentes: bus de données, bus d'adresse et bus de control. L'organisation entre ces trois composants et leur disposition définit une architecture dans notre cas on se limite à deux modèles: Modèle de Von Neumann et *Harvard*



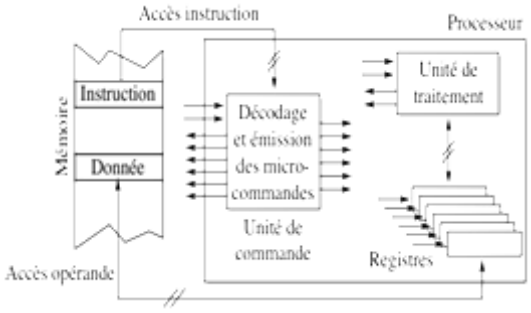


Le processeur, la mémoire et les entrées/sorties (I/O) sont reliés par les trois bus
Dans le cas de Harvard la mémoire est divisée en deux parties celle qui contient le code (l'ordre à exécuter) et celle qui contient les données et ici, on doit bien distinguer la différence: une opération telle que l'addition ne peut s'effectue que si on a des données exemple s= a+b
Sans données l'opération ne s'effectue pas donc l'ordre à exécuter c'est l'addition (+) et les données sont a et b.
Par instruction on sous entend opération est données

Un programme (le code + données) est un ensemble finie d'instructions (d'opérations et de données) qui réalisent un traitement bien précis. Ce programme sera exécuté par le microprocesseur pour donner le résultat du traitement ciblé.
Dans l'architecture Harvard les deux composantes de l'instruction sont placées dans deux mémoires différentes dont le but est d'accélérer le traitement par ce qu'on appelle le pipelining (traitement parallèle)

Architecture de L'unité centrale ou le µP

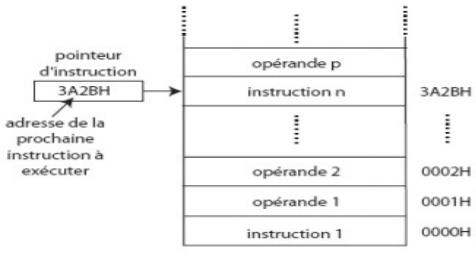
D'une manière générale l'architecture interne d'une CPU est composée d'une ALU d'une unité de contrôle et de registres.
La CPU charge, analyse et exécute les instructions présentes en mémoire de façon séquentielle, c'est-à-dire une instruction à la suite de l'autre. Elle contient une unité de calculs arithmétiques et logiques (UAL ou ALU en anglais), d'un bus interne, d'un bus externe se connectant au système, d'un décodeur d'instructions qui décode l'instruction en cours, et des registres pour mémoriser des résultats temporairement.
Ce sont les registres qui permettent la communication entre le programme (résidant en mémoire) et la CPU. Ils sont 'l'interface' de la CPU. En effet, pratiquement, toutes les données qui passent par la CPU, pour être traitées par celle-ci, doivent se trouver dans les registres de cette dernière.



Il existe différents types de registres dans une CPU: les registres de traitements, les registres d'adressages et les registres d'état. Les Registres de traitement sont des registres destinés au traitement des valeurs contenues dans celle-ci; par exemple on peut effectuer une addition d'un registre de traitement avec un autre registre, effectuer des multiplications ou des traitements logiques. Les registres d'adressage permettent de pointer un endroit de la mémoire; ils sont utilisés pour lire ou écrire dans la mémoire. Les registres d'état (ou drapeau: FLAG en anglais) indiquant l'état du processeur et 'le résultat' de la dernière instruction exécutée. Les plus courants sont le Zero Flag (ZF) qui indique que le résultat de la dernière opération est égale a zéro (après une soustraction par exemple), le Carry Flag (CF) qui indique qu'il y a une retenue sur la dernière opération effectuée, Overflow Flag (OF) qui indique un dépassement de capacité de registre, etc.

Registres PC et IR

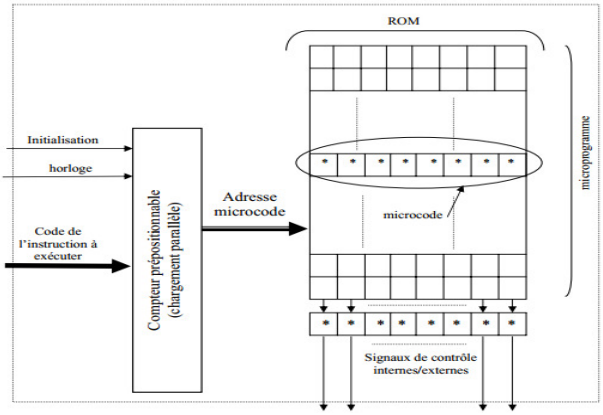
Pour exécuter les instructions dans l'ordre établi par le programme, le microprocesseur doit savoir à chaque instant



l'adresse de la prochaine instruction à exécuter. Le µP utilise un registre contenant cette information. Ce registre est appelé pointeur d'instruction (IP : Instruction Pointer) ou compteur d'instructions, compteur ordinal ou Le compteur de programme (PC pour Program counter). Le PC contient en permanence l'adresse de la prochaine instruction à exécuter. À chaque début de cycle d'exécution, l'instruction à exécuter est chargée dans le registre IR à partir de l'adresse contenue dans le registre PC. Ensuite, le registre PC est incrémenté pour pointer sur l'instruction suivante.
Le registre d'instruction(IR) contient l'instruction en cours d'exécution. Ce registre est chargé au début du cycle d'exécution par l'instruction dont l'adresse est donnée par le PC.

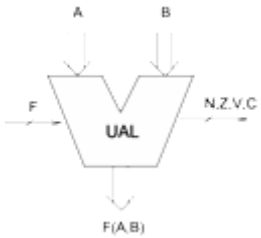
L'unité de décodage d'instructions

Appelée aussi unité de commande a pour rôle de générer en séquence les différents signaux de fonctionnement internes et externes au µP correspondant à l'instruction à traiter. Le principe peut être vu à partir de la figure suivante :
Le code opératoire est décodé et des signaux de commande pour l'UAL sont produits en fonction de l'opération demandée qui est alors exécutée.
Remarque : pour exécuter une instruction, l'UAL utilise des registres de travail.



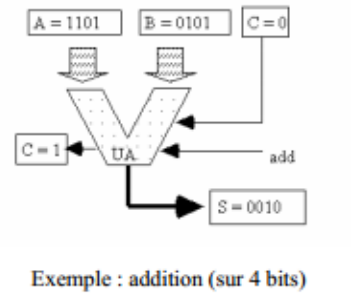
L'unité arithmétique et logique (UAL ou ALU)

Dans un processeur, on regroupe souvent dans la même unité fonctionnelle les différents opérateurs d'arithmétique entière (additionneurs,multiplieurs,etc.),les opérateurs de logique booléenne (AND, OR, etc.) et les opérations de décalage et de rotation de bits.
On appelle un tel module unité arithmétique et logique (UAL), et elle a la forme de la figure suivante :



Les opérations que réalise l'UAL ont une parité de 2, et parfois 1. Les opérandes sont présentés sur deux bus A et B de même largeur ; dans le *schéma de L'ALU on* indique sur F le code d'une opération à effectuer (par exemple, une addition), et le résultat est disponible sur le bus de sortie F(A,B) après un certain temps de propagation des calculs. Les flags ou indicateurs N,Z,V,C donnent des informations sur la nature du résultat de l'opération. N indique que le résultat est négatif, Z indique qu'il est nul, V indique un débordement et C indique la présence d'une retenue.

Il faut noter que l'UAL est un opérateur combinatoire :il n'a pas d'horloge, et le résultat d'une opération n'est disponible qu'après un temps variable (mais borné)



Exemple : addition (sur 4 bits)

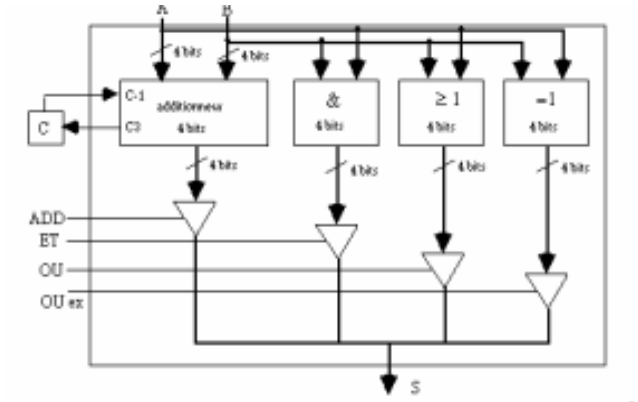


Figure : réalisation matérielle de L ALU (addition,AND,OR,XOR)

Horloge du processeur

Le temps d'exécution propre à chaque instruction, est exprimé en cycles de l'horloge interne qui cadence l'activité du processeur.

• envoie des impulsions régulières au processeur

• a chaque nouvelle impulsion µP exécute une opération élémentaire

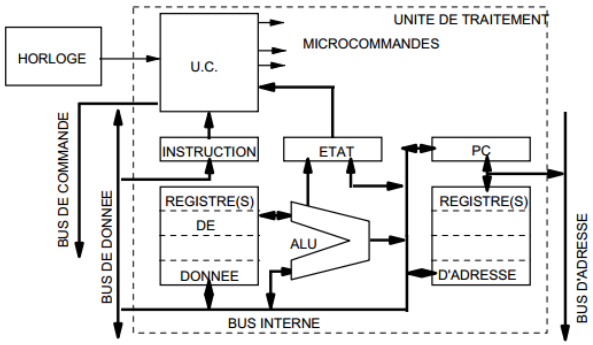
Les registres

Les registres sont réalisés à partir de bascules type D ayant des sorties « haute impédance ». Leur rôle est de mémoriser (stocker) à des instants précis les données qui circulent sur le bus interne au microprocesseur. Les signaux qui contrôlent les registres (horloge, commande « haute impédance ») sont générés par l'unité de décodage d'instruction.

Les amplificateurs de ligne

Ces circuits sont aussi appelés des « buffers ». Ils ont pour rôle de fournir le courant nécessaire à la charge que représente le bus. Ils disposent en général d'une commande qui permet de mettre leur sortie à haute impédance (Hi-Z). Lorsque les bus sont de type bidirectionnels, une commande permet de choisir le sens de transfert des données. Un décodeur peut être utilisé pour n'activer qu'au plus un composant 3-états à la fois, puisqu'au plus une seule de ses sorties est active à la fois

Cycle d'exécution d'une instruction



L'exécution d'une instruction passe par plusieurs phase généralement appelées microinstructions. On discute dans ce qui suit ces étapes en prenant l'exemple de l'instruction s= a+b.

Fetch (apporter l'instruction de la mémoire vers le processeur sous sa forme binaire obtenue par compilation ou assemblage d'un programme écrit par l'utilisateur)

Décodage de l'instruction : l'instruction contient l'information qui spécifie l'opération en question à effectuer sous forme d'un code binaire appelé code opératoire ou opcode qu'il faut interpréter (décoder) pour orienter le processeur à activer le circuit approprié (ici addition +) qui réalise cette opération

Fetch des données (apporter les données nécessaires de la mémoire vers le processeur ici a et b)

Exécution réalisation effective de l'opération incluse dans l'instruction (activer l'additionneur qui donne le résultat s)

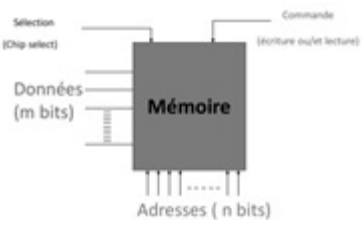
Stockage du résultat dans la mémoire (ici s)

Ces Cinq cycles d'exécution de l'instruction s'exécutent séquentiellement mais on peut les chevaucher et réaliser un traitement en parallèle (notion de pipe).

On remarque qu'en moyenne l'accès à la mémoire se fait trois fois lors de l'exécution d'une instruction comparativement au traitement interne par le processeur d'où on déduit que la vitesse d'exécution d'une instruction prend en considération la vitesse de traitement du microprocesseur et la vitesse d'accès à la mémoire et c'est pour cela qu'on mesure la vitesse de traitement par l'unité MIPS (**m**illion **i**nstruction **p**er **S**econd)

La mémoire

La mémoire centrale, essentielle au fonctionnement du microprocesseur, est réalisée à base de semi-conducteurs.

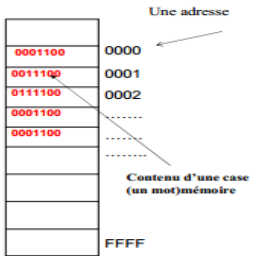


• C' est une mémoire vive : accès en lecture et écriture et elle a besoin d'une alimentation pour fonctionner

• elle est dite à accès aléatoire (RAM : Random Acces Memory) c'est-à-dire que le temps d'accès à l'information est indépendant de sa place en mémoire.

• La mémoire est un composant logique séquentiel caractérisé par sa possibilité de mémoriser et stocker l'information : L'information

stockée (données) est placées dans des cellules de mémorisations (bascule D) ces cellules (appelées aussi des cases mémoires) sont repérées par des adresses. Une adresse à n bits donne au maximum 2^n cases mémoires ; d'une manière générale une case mémoire contient 8 bascules pouvant stocker 8 bits (ici données m = 8 bits) en plus des données et des adresses, la mémoire contient aussi des lignes de contrôle tel que (CS) Chip select (sélection de la puce mémoire quand CS=1) et la ligne d'écriture RD quand on veut lire de la mémoire (RD=1) et la ligne WR write pour écrire dans la mémoire (WR=1)



La mémoire centrale peut être vue comme un large vecteur (tableau) de mots ou octets.

•Une adresse est un numéro unique qui permet d'accéder à un mot mémoire.

•Les adresses sont séquentielles (consécutives)

•La taille de l'adresse (le nombre de bits) dépend de la capacité de la mémoire.

• Nombre de cases mémoires est Le nombre de données pouvant être stockées est $N = 2^n$ (où n est le nombre de bits d 'adresse)

La capacité d'une mémoire $C = m \times N$

Exemple : Calcul de la capacité d'une mémoire

Mémoire : 16 bits d'adresse et 8 bits de données: m = 8 Nombre de cases : $N = 2^{16} = 65536$

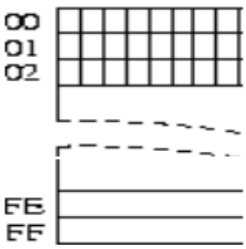
Capacité : $C = m \times N = 8 \times 65536 = 524288$ bits

Pour des raisons de simplification, on exprime la capacité :

• en kilo-octets (Ko) 1 octet : 8 bits 1 Ko = 2^{10} = 1024 octets

• en méga-octets (Mo) 1 Mo = 2^{20} octets

• en giga-octets (Go) 1 Go = 2^{30} octets



Exercice:

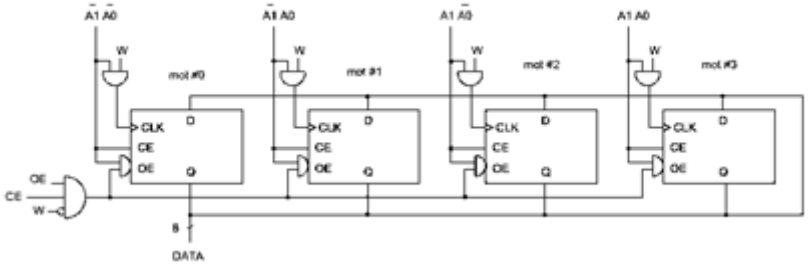
Les adresses varient de 00 a FF en hexadécimal et chaque adresse pointe vers une case a 8 bits de données, déterminer la capacité de cette mémoire

Réponse: $2^8 \times 8 = 256$ octets ou $2^8 \times 2^3 = 2^{11} = 2048$ bits

Structure interne d'une mémoire statique asynchrone

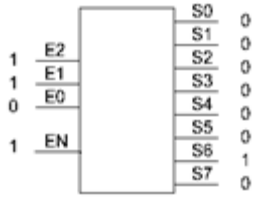
La figure montre l'Implémentation d'une RAM statique asynchrone de 4 mots de 8 bits. Le nombre de ligne d'adresses=n=$\log_2 N = \log_2 4 = 2$ on les représente par $A_1 A_0$

Ajouter une ligne d'adresse revient à multiplier par deux le nombre de colonnes, tout en gardant la même structure générale.



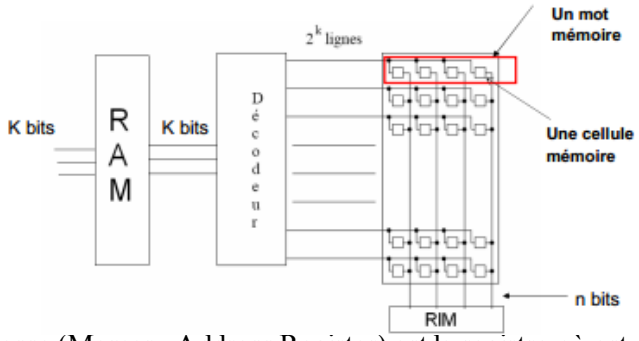
Décodage d'adresse

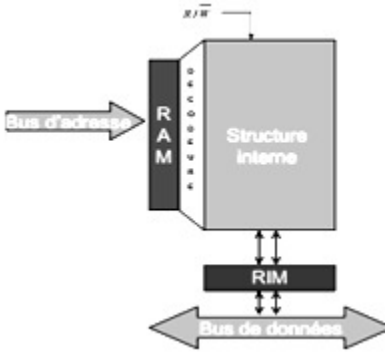
Il a pour but d'assigner à chaque bloc mémoire et aux différents périphériques qui composent le système un emplacement et un espace dans le champ adressable par le microprocesseur.



Un décodeur est un circuit possédant n entrées et 2^n sorties, à tout moment, une et une seule sortie est active : celle dont le numéro correspond à la valeur binaire présente sur les n entrées. Le décodeur traduit donc la valeur d'entrée en une information de position spatiale. Exemple de la figure Décodeur 3 vers 8. Avec la valeur 6 en entrée (110 en binaire),la sortie numéro 6 est activée.

•Lorsque une adresse est chargée dans le registre RAM , le décodeur va recevoir la même information que celle du RAM.
•A la sortie du décodeur nous allons avoir une seule sortie qui est active Cette sortie va nous permettre de sélectionner un seule mot mémoire (pointe vers une seule case).





Le registre d'adresse (Memory Address Register) est le registre où est stockée l'adresse de la case mémoire lue ou écrite lors d'un accès à la mémoire. La donnée transite par le registre de donnée (Memory Data Register). Lors d'une lecture, la donnée parvient au processeur dans ce registre. Lors d'une écriture, la donnée est placée dans ce registre par le processeur avant d'être écrite dans la case mémoire.

Le bus de données

Ce sont des lignes physiques (équipotentielles) qui véhiculent les données. Ce bus est bidirectionnel (les données peuvent entrer ou sortir du µP suivant qu'il est en mode lecture ou écriture). La taille de ce bus correspond à la taille du bus de données interne au µP. La largeur du bus de données détermine la quantité de données que le processeur peut lire ou écrire dans une mémoire ou un cycle d'E / S (cycle machine)

le bus d'adresses

Ce sont des lignes physiques (équipotentielles) qui véhiculent les adresses générées par le microprocesseur. La taille du bus fixe sa capacité d'adressage. Dans les systèmes monoprocesseurs et selon leurs types, le bus d'adresses est souvent unidirectionnel.

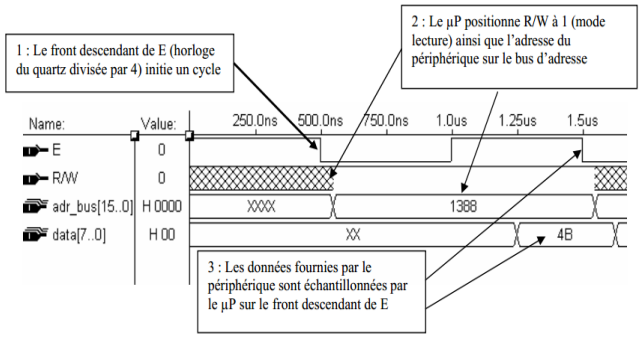
Le bus de contrôle

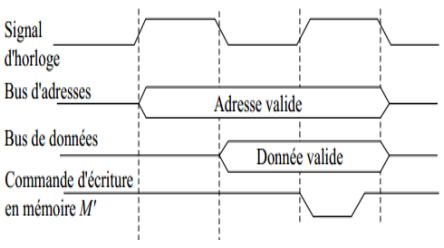
Il est de taille variable suivant la complexité du processeur. C'est lui qui synchronise les échanges entre le processeur et ses périphériques. Le chronogrammeci-dessous illustre un cycle de lecture d'un périphérique du microprocesseur Motorola 6809.

Golden rule

"On ne met rien dans le bus d'adresse ou de données jusqu'à ce que le bus de contrôle ordonne de le faire."

Ceci se traduit par la synchronisation des différents bus à l'aide du signal d'horloge comme c'est illustré dans le chronogramme suivant. Notez les étapes ordonnées 1,2 et 3 qui se font en synchronisme avec l'impulsion d'horloge.





Pour lire une information de la mémoire il faut effectuer les opérations suivantes:
– Charger dans le registre RAM l'adresse du mot à lire.
– Lancer la commande de lecture (R/W=1)
– L'information est disponible dans le registre RIM au bout d'un certain temps (temps d'accès)
Pour écrire une information dans la mémoire il faut effectuer les opérations suivantes:
– Charger dans le RAM l'adresse du mot ou se fera l'écriture.
– Placer dans le RIM l'information à écrire.
– Lancer la commande d'écriture pour transférer le contenu du RIM dans la mémoire

On utilise très souvent, Pour les signaux de contrôle RD,WR,CS etc., la logique négative qui permet d'activer une ligne par le niveau logique bas (0) et la désactiver avec le niveau haut (1) ceci est due à des économies de consommation énergétique. Exemple: CS=0 le boîtier est sélectionné CS=1 le boîtier n'est pas sélectionné.

Exercice : On veut réaliser une mémoire de capacité C , mais nous disposons uniquement de boîtiers de taille inférieur ?
• Soit M une mémoire de capacité C , tel que m est le nombre de mot et n la taille d'un mot.
• Soit M' un boîtier de capacité C' , tel que m' le nombre de mot et n' la taille d'un mot.
• On suppose que C > C' (m >= m' , n >=n')
• Quel est le nombre de boîtiers M' nécessaire pour réaliser la mémoire M ?
• Pour connaître le nombre de boîtiers nécessaire, il faut calculer les deux facteurs suivants :
P =m/m' et Q=n/n
On veut réaliser une mémoire de 1Ko (la taille d'un mot est de 16 bits) en utilisant des boîtiers de taille 1Ko mots de 4 bits) ?
• Solution :
(m,n)=(1024,16) taille du bus d'adresses est de 10 bits (A9…A0), taille de bus de données est du 16 bits (D15….D0)
(m',n')=(1024,4) taille du bus d'adresses est de 10 bits (A9'…A0'), taille de bus de données est du 4 bits (D3'….D0')
• P=1024/1024=1 (extension lignes)
• Q=16/4=4 (extension colonnes)
• Le nombre total de boîtiers P.Q=4

Architectures RISC et CISC

Les premières générations de microprocesseurs étaient à architecture CISC (Complex Instructions Set Computer) ce qui nécessitait plusieurs cycles d'horloge de base (fournie par le quartz) pour exécuter une instruction. Au fil du temps les concepteurs de circuits se sont aperçus que 80% des traitements effectués faisaient appel à 20% des instructions du µP. L'idée est donc venue de développer des microprocesseurs possédant un jeu d'instructions réduit (RISC : Reduced Instruction Set Computer) mais exécutable dans un temps très court (souvent en un seul cycle d'horloge). Ces microprocesseurs sont souvent à architecture de « Harvard » et disposent d'un mode de fonctionnement en « pipeline ».
Aujourd'hui les deux types de processeurs cohabitent et leur utilisation est fonction des applications visées (station de travail, serveur, système temps réel …).
Notons toutefois que si les architectures RISC sont plus simples à réaliser sur le plan matériel, les programmes assembleurs et les compilateurs associés sont plus compliqués à réaliser.

	RISC	CISC
Avantages	Temps d'exécution court, hardware plus simple, coût puce électronique réduit	Instructions puissantes, programme plus simple à écrire, moins de lignes de programme, compilateur simple à concevoir
Inconvénients	Programme plus complexe à écrire, plus de lignes, compilateur difficile à concevoir	Temps exécution long, hardware plus compliqué et plus coûteux