Ministère de l'Enseignement Supérieur, de la Recherche Scientifique et de la Technologie

Université Virtuelle de Tunis

Microprocesseurs et Microcontrôleurs

LES SYSTEMES MICRO-PROGAMMES

Jlassi Khaled

I. Mise en situation.

Un système numérique, intégrant de l'électronique, fait souvent apparaître des fonctions ayant pour rôle le traitement d'informations : opérations arithmétiques (addition, multiplication...) ou logiques (ET, OU...) entre plusieurs signaux d'entrée permettant de générer des signaux de sortie. Ces fonctions peuvent être réalisées par des circuits intégrés analogiques ou logiques. Mais, lorsque le système devient complexe, et qu'il est alors nécessaire de réaliser un ensemble important de traitements d'informations, il devient plus simple de faire appel à une structure à base de microcontrôleur ou microprocesseur. Le développement de ces composants programmables a été rendu possible grâce l'essor considérable qu'a connu la microélectronique et notamment les techniques d'intégration. Cette évolution a permis en 1971, la fabrication du premier microprocesseur par la société INTEL. Ce microprocesseur, le « 4004 », comportait déjà 2300 transistors et fonctionnait avec un bus de données de 4 bits. Depuis, l'intégration du nombre de transistors dans les microprocesseurs n'a cessé d'évoluer, parallèlement à la puissance de calcul et la rapidité d'exécution. Aujourd'hui, un microprocesseur Pentium IV comporte a peu près 24 millions de transistors et peut traiter des données de 8, 16, 32, 64 bits en même temps. La puissance des microprocesseurs d'aujourd'hui a orientée leur utilisations vers le traitement des informations de masse (Gestion d'une base de donnée, Gestion des périphériques bloc, …), le calcul scientifique ainsi que tout ce qui est interface homme machine réactif (clavier, souris, écran, …). Comme nous pouvons le constater, le domaine d'application des microprocesseurs reste vaste. C'est pourquoi nous les classons dans la catégorie des composants programmables généralistes, cela signifie qu'ils peuvent tout faire, mais ils ne sont optimisés pour rien. La majorité des microprocesseurs ont une architecture CISC (*Complex Instruction Set Computer*, ce qui signifie "ordinateur avec jeu d'instructions complexes". C'est le cas des processeurs de type x86, c'est-à-dire les processeurs fabriqués par Intel, AMD, Cyrix, ... Les processeurs basés sur l'architecture CISC peuvent traiter des instructions complexes, qui sont directement câblées sur leurs circuits électroniques, c'est-à-dire que certaines instructions difficiles à créer à partir des instructions de base sont directement imprimées sur le silicium de la puce afin de gagner en rapidité d'exécution. L'inconvénient de ce type d'architecture provient justement du fait que des fonctions supplémentaires sont imprimées sur le silicium, d'où un coût élevé. D'autre part, les instructions sont de longueurs variables et peuvent parfois prendre plus d'un cycle d'horloge ce qui les rend lentes à l'exécution. Néanmoins, avec la considérable augmentation de la taille des puces électroniques et la gigantesque accélération des fréquences d'horloge, la

puissance de calcul d'un microprocesseur CISC d'aujourd'hui est considérable. Son caractère généraliste lui permet d'être par excellence le composant de base de l'informatique. Mais en instrumentation et automatisme on lui préférera généralement des composants plus spécialisés ne nécessitant pas, ni un calcul complexe ni un traitement d'informations de masse. C'est pourquoi dans les applications industrielles, que ce soit d'automatisme ou d'instrumentation, le microcontrôleur est le composant programmable le plus utilisé. Il comporte sur sa puce un certain nombre d'interfaces qui n'existent pas sur un microprocesseur, par contre il est généralement moins puissant en terme de rapidité ou de taille de mémoire adressable et le plus souvent cantonné aux données de 8 ou 16 bits. Les microcontrôleurs utilisent la technologie RISC (Reduced Instruction Set Computer), dont la traduction est "ordinateur à jeu d'instructions réduit" et n'a pas de fonctions supplémentaires câblées. Ce qui qui implique une programmation plus difficile et un compilateur plus puissant. Les instructions d'un microcontrôleur sont tellement peu nombreuses (en moyenne une soixantaine) qu'il est possible de les graver directement sur le silicium sans alourdir de manière dramatique leur fabrication. L'avantage d'une telle architecture est bien évidemment le coût réduit au niveau de la fabrication des processeurs l'utilisant. De plus, les instructions, étant simples, ils sont exécutés en un cycle d'horloge, ce qui rend l'exécution des programmes plus rapides qu'avec des processeurs basés sur une architecture CISC. En plus, de tels processeurs sont capables de traiter plusieurs instructions simultanément en les traitant en parallèle. Les microcontrôleurs ont permis de faire évoluer les systèmes micro programmés vers encore plus de simplicité et de rapidité. Ils sont aujourd'hui utilisés dans la plupart des réalisations industrielles grand public ou professionnelles, ils gèrent au plus juste et au plus vite les applications. Leur évolution a permis l'intégration de circuits complexes variés. Ces circuits ont été intégrés sur une même puce donnant ainsi beaucoup de flexibilité et de puissance de commande au microcontrôleur. Cette polyvalence lui permet d'occuper une place importante que ce soit en instrumentation, en commande ou en automatisme industriel. Le meilleur exemple est bien évidemment les automates programmables qui sont tous équipés de microcontrôleurs.

II. Description et structure interne d'un microcontrôleur.

Un microcontrôleur est un composant réunissant sur un seul et même silicium un micro-processeur, divers dispositifs d'entrées/sorties et de contrôle d'interruptions ainsi que de la mémoire, notamment pour stocker le programme d'application. Dédié au contrôle, il embarque également un certain nombre de périphériques spécifiques

des domaines ciblés (bus série, interface parallèle, convertisseur analogique numérique, ...). Les microcontrôleurs améliorent l'intégration et le coût (lié à la conception et à la réalisation) d'un système à base de microprocesseur en rassemblant ces éléments essentiels dans un seul circuit intégré. On parle alors de "système sur une puce" (en anglais : "System On chip"). Il existe plusieurs familles de microcontrôleurs, se différenciant par la vitesse de leur processeur et par le nombre de périphériques qui les composent. Toutes ces familles ont un point commun c'est de réunir tous les éléments essentiels d'une structure à base de microprocesseur sur une même puce. Voici généralement ce que l'on trouve à l'intérieur d'un tel composant :

- ☞ Un microprocesseur (C.P.U.),
- ☞ Des bus,
- ☞ De la mémoire de donnée (RAM et EEPROM),
- ☞ De la mémoire programme (ROM, OTPROM, UVPROM ou EEPROM),
- ☞ Des interfaces parallèles pour la connexion des entrées / sorties,
- ☞ Des interfaces séries (synchrone ou asynchrone) pour le dialogue avec d'autres unités,
- ☞ Des timers pour générer ou mesurer des signaux avec une grande précision temporelle.

Sur la figure (2.1) nous présentons le schéma type d'un microcontrôleur.

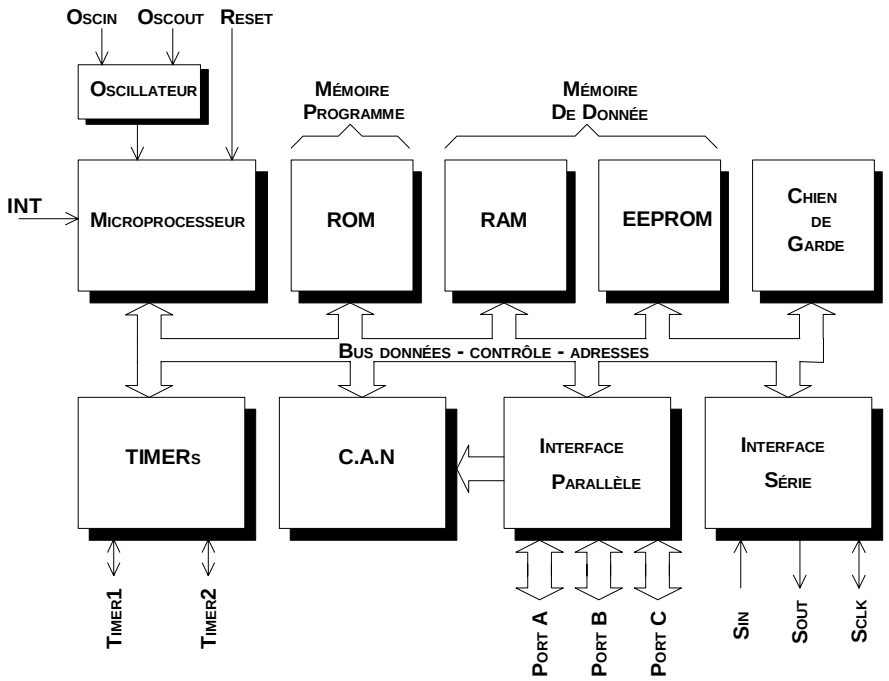


Figure 2.1. Schéma type de tout appareil programmable.

La présence de ces divers éléments de base est indispensable et, même s'ils ne sont pas toujours aussi visibles que sur notre synoptique, ils sont toujours présents. Ces éléments sont les mêmes que pour un système informatique classique mais, dans le cadre d'une application pouvant être traitée par un microcontrôleur, leurs tailles et fonctions différent un peu de ce que nous avons peut-être l'habitude de voir. L'unité centrale, généralement constituée par un microprocesseur plus ou moins évolué, exécute le programme qui va donner vie à l'application. Pour les applications industrielles, ces programmes ont en commun le fait qu'ils ne nécessitent que très rarement des calculs complexes alors qu'ils sont très friands de manipulations d'informations d'entrées/sorties. C'est pour cette raison que la majorité des microcontrôleurs inclut une interface parallèle de type PIA(Peripheral Interface Adapter figure (2.1)). Etant donné la nature des applications des microcontrôleurs souvent les programmes sont contenus dans le deuxième élément de la figure (2.1) qui est la mémoire morte ou ROM. Cette mémoire peut-être constituée de diverses façons: mémoire programmée par masque, mémoire UVPROM ou EEPROM. De cette façon, nous n'avons plus besoin de charger le programme d'application à chaque mise sous tension. L'exécution de l'application est enclenchée

automatiquement à la mise sous tension, ce qui n'est pas le cas pour les systèmes à base de microprocesseur qui nécessitent un programme de démarrage (système d'exploitation ou moniteur). Pour pouvoir travailler correctement notre microprocesseur a souvent besoin de stocker des données temporaires quelque part et c'est là qu'intervient la mémoire vive ou RAM qui, contrairement aux systèmes informatiques classiques, peut être de très petite taille. Le dernier élément qui ne manque pas d'importance dans les applications susceptible de faire appel à un microcontrôleur est tout ce qui concerne les circuits d'interface avec le monde extérieur. Contrairement aux systèmes informatiques classiques où les interfaces sont bien connues, de types assez peu diversifiés et en nombre relativement limité; les interfaces que nous pouvons rencontrer dans des applications industrielles sont absolument quelconques. Il y a en effet assez peu de points communs entre un moteur pas à pas, un afficheur à cristaux liquides ou bien encore un programmateur de machine à laver. Le schéma type d'un microcontrôleur est issu d'une analyse assez forte des divers systèmes de commande et d'automatisme réalisés avant l'avènement des microcontrôleurs. Les fabricants de circuits intégrés ont affinés un peu la définition de ce qu'il fallait intégrer pour arriver à un schéma-type analogue à celui de la figure (1.1). Nous y retrouvons bien évidemment un microprocesseur, généralement simplifié et avec un jeu d'instructions réduit mais auquel des instructions de manipulation de bits, très utiles pour faire des entrées/sorties, lui ont été ajoutées. Dans certains cas, ce microprocesseur se voit doté d'un très grand nombre de registres internes qui servent alors de mémoire vive. Ce qui évite l'utilisation d'une mémoire externe pour le stockage des données et rend le système encore plus compact. Dans un grand nombre de microcontrôleurs, nous trouvons également de la mémoire morte intégré. Avec les progrès technologiques les fabricants ont appris à placer sur la puce de la mémoire programmable électriquement et effaçable aux ultraviolets (UVPROM) ou, plus récemment encore, de la mémoire programmable et effaçable électriquement (EEPROM). On trouve donc à l'heure actuelle au moins quatre types différents de microcontrôleurs :

- ceux sans aucune mémoire morte ;

- ceux avec EPROM (programmable et effaçable à l'ultra violet);

-ceux avec EEPROM ;

-ceux avec un mélange de ces combinaisons (ROM -EEPROM ou UVPROM - EEPROM).

Pour ce qui est de la mémoire vive ou RAM, la situation est plus simple. Quasiment tous les microcontrôleurs disposent d'une RAM interne de taille en principe assez faible et, lorsqu'elle n'est pas explicitement visible sur le synoptique, c'est que l'unité centrale dispose d'assez de registres pour servir de RAM comme nous l'avons expliqué précédemment. Il est un peu plus délicat de faire un schéma type au niveau des circuits d'interface car c'est là que les différents microcontrôleurs se distinguent en fonction des créneaux d'applications qu'ils visent. Néanmoins, on rencontre généralement les éléments de base suivants :

- des lignes d'entrées/sorties parallèles en nombre variable selon la vocation et la taille du boîtier (un problème de nombre maximum de pattes se posant très vite avec l'accroissement du nombre de ces lignes);

- au moins une interface d'entrée/sortie série asynchrone, plus ou moins évoluée selon les circuits;

- un ou plusieurs timers internes dont les possibilités peuvent être très variables mais qui fonctionnent généralement en compteurs, décompteurs, générateurs d'impulsions programmables, etc. ;

- parfois, mais c'est un peu plus rare, un ou des convertisseurs analogiques numériques précédés ou non de multiplexeurs pour offrir plusieurs voies ;

- parfois aussi, mais c'est également plus rare, un convertisseur numérique analogique.

Enfin, bien que ce ne soit pas une véritable interface d'entrée/sortie au sens où nous l'entendons, certains microcontrôleurs disposent d'un accès à leur bus interne. Ceci permet de connecter des boîtiers destinés à accomplir des fonctions qui font défaut sur la puce ce qui est parfois utile. Précisons, mais c'est une évidence, que tous les microcontrôleurs sans mémoire morte interne disposent nécessairement de cette interface puisqu'il faut impérativement leur permettre d'accéder à une mémoire morte externe contenant le programme de l'application. Bien sûr, notre schéma-type ne peut recouvrir tous les cas mais il vous donne une idée assez précise de ce que contient un microcontrôleur. Il est bien évident d'après ce schéma que la puissance d'un microcontrôleur est directement liée au processeur qu'il intègre. C'est

pourquoi les constructeurs développent souvent un cœur de processeur destiné aussi bien à décliner une gamme de microprocesseurs que de microcontrôleurs ; La tendance aujourd'hui et un cœur de processeur 16 ou 32 bits qui représente une augmentation de la surface occupée sur le silicium de seulement quelques pour-cent par rapport à un circuit en 8 bits. Le prix du microcontrôleur étant directement lié à sa taille, embarquer une puissance supérieure est un choix qui ne grève plus le budget. Opter pour un cœur de processeur en 16 ou en 32 bits, c'est permettre entre autres des exécutions parallèles, un espace mémoire élargi, des interfaces de communications ou encore le remplacement de fonctions analogiques par des traitements numériques. Le cœur de processeur 32 bits le plus prisé aujourd'hui est l'ARM7. Pour preuve, plusieurs fabricants l'utilisent aujourd'hui pour le développement de microcontrôleurs 32 bits. Les secteurs les plus visés sont surtout les systèmes de communication sans fil, les téléphones portables et l'industrie automobile. Par exemple les microcontrôleurs de la famille STR7 de STMicroelectronics sont basés sur le cœur ARM7. Ces microcontrôleurs intègrent une multitude d'interfaces série (CAN, USB, I2C, SPI,..), de la mémoire SDRAM, de la mémoire FLASH, des convertisseurs analogiques numériques, 48 entrées/sorties, une interface JTAG, plusieurs horloges temps réel,.. L étude d'un microcontrôleur basé sur le cœur ARM7 comme le STR7 dépasse largement le cadre de ce cours. La diversité des périphériques qu'il intègre rend sa compréhension très difficile c'est pourquoi nous préférons présenter un cœur de processeur beaucoup plus simple comme exemple d'étude dans le cadre de ce cours. Dans le paragraphe suivant nous allons présenter les éléments généraux communs aux divers types de processeurs.

III. Le processeur.

Le processeur (microprocesseur) est le composant hardware le plus connu d'un système micro-programmé. C'est l'unité intelligente de traitement des informations. Son travail consiste à lire des programmes (des suites d'instructions), à les décoder et à les exécuter. Les années 80 voyaient l'émergence de ces circuits avec les Zylog Z80, 6800 de Motorola, le 8085 de Intel qui est souvent utilisé en tant que microcontrôleur. Avec l'arrivée des PC-XT d'IBM et l'utilisation du 8088, INTEL devenait maître du marché fin des années 80. L'interfaçage du processeur avec l'extérieur nécessite 3 bus: un bus de données, un bus d'adresse et un bus de commande. Il existe des processeurs basés sur l'architecture CISC et d'autres basés sur l'architecture RISC. Cependant certains processeurs sont difficilement classifiables comme le CPU i486 également appelé 80486. Ce véritable processeur 32 bits (bus internes et externes) est à mi chemin entre le tout CISC et le tout RISC. Ce processeur offre des performances similaires aux processeurs RISC 32 bits, tout en restant compatible 100% avec son

prédécesseur CISC le 80386. Par contre le processeur 68040 de Motorola est basé sur l'architecture CISC. Il est présenté par Motorola comme étant un processeur CISC pur et dur. Les premiers concepteurs de processeurs rajoutaient le plus d'instructions possibles pour permettre à l'utilisateur de peaufiner ses programmes. Néanmoins, ces multiples instructions ralentissent le fonctionnement du microprocesseur et sont peu utilisées en pratique. Actuellement, on utilise de plus en plus de processeurs **RISC** (Reduced Instruction Set Computer). Le nombre d'instructions est réduit, mais exécutées nettement plus rapidement. Chaque instruction complexe peut être programmée par plusieurs instructions simples. Un processeur est constitué de:

- une unité de commande qui lit les instructions et les décode;
- une unité de traitement (UAL - unité arithmétique et logique) qui exécute les instructions;
- d'un ensemble de mémoire appelés registres;
- d'un bus de données externe;
- d'un bus d'adresse externe;
- d'un bus de commande externe;
- d'un bus de données interne reliant l'unité de commande l'UAL et les registres.

Lorsque tous ces éléments sont regroupés sur une même puce, on parle alors de microprocesseur. La figure ci dessous donne une idée sur l'architecture interne d'un microprocesseur. Sur cette figure nous pouvons voir les 3 bus qui permettent au microprocesseur de communiquer avec l'extérieur.

III.1 Architecture de base d'un microprocesseur

III.1.1 L'unité de commande.

Elle permet de "séquencer" le déroulement des instructions. Elle effectue la recherche en mémoire de l'instruction, le décodage, l'exécution et la préparation de l'instruction suivante. L'unité de commande élabore tous les signaux de synchronisation internes ou externes (bus des commandes) au microprocesseur.

III.1.2 L'unité arithmétique et logique (UAL).

C'est l'organe qui effectue les opérations:

Arithmétiques : addition, soustraction, multiplication, ...

Logiques : et, ou, non, décalage, rotation, ...

Deux registres sont associés à l'UAL : l'accumulateur et le registre d'état.

III.1.2.1 L'accumulateur (nommé : A).

C'est une des deux entrées de l'UAL. Il est impliqué dans presque toutes les opérations réalisées par l'UAL. Certains constructeurs ont des microprocesseurs à deux accumulateurs (Motorola : 6800).

Exemple: **A** étant l'accumulateur et **B** un registre, on peut avoir : A+B (ADD A,B : addition du contenu du registre A avec celui du registre B, le résultat étant mis dans A)

I. Mise en situation.

Un système numérique, intégrant de l'électronique, fait souvent apparaître des fonctions ayant pour rôle le traitement d'informations : opérations arithmétiques (addition, multiplication...) ou logiques (ET, OU...) entre plusieurs signaux d'entrée permettant de générer des signaux de sortie. Ces fonctions peuvent être réalisées par des circuits intégrés analogiques ou logiques. Mais, lorsque le système devient complexe, et qu'il est alors nécessaire de réaliser un ensemble important de traitements d'informations, il devient plus simple de faire appel à une structure à base de microcontrôleur ou microprocesseur. Le développement de ces composants programmables a été rendu possible grâce l'essor considérable qu'a connu la microélectronique et notamment les techniques d'intégration. Cette évolution a permis en 1971, la fabrication du premier microprocesseur par la société INTEL. Ce microprocesseur, le « 4004 », comportait déjà 2300 transistors et fonctionnait avec un bus de données de 4 bits. Depuis, l'intégration du nombre de transistors dans les microprocesseurs n'a cessé d'évoluer, parallèlement à la puissance de calcul et la rapidité d'exécution. Aujourd'hui, un microprocesseur Pentium IV comporte a peu près 24 millions de transistors et peut traiter des données de 8, 16, 32, 64 bits en même temps. La puissance des microprocesseurs d'aujourd'hui a orientée leur utilisations vers le traitement des informations de masse (Gestion d'une base de donnée, Gestion des périphériques bloc, …), le calcul scientifique ainsi que tout ce qui est interface homme machine réactif (clavier, souris, écran, …). Comme nous pouvons le constater, le domaine d'application des microprocesseurs reste vaste. C'est pourquoi nous les classons dans la catégorie des composants programmables généralistes, cela signifie qu'ils peuvent tout faire, mais ils ne sont optimisés pour rien. La majorité des

microprocesseurs ont une architecture CISC (*Complex Instruction Set Computer*, ce qui signifie "ordinateur avec jeu d'instructions complexes". C'est le cas des processeurs de type x86, c'est-à-dire les processeurs fabriqués par Intel, AMD, Cyrix, ... Les processeurs basés sur l'architecture CISC peuvent traiter des instructions complexes, qui sont directement câblées sur leurs circuits électroniques, c'est-à-dire que certaines instructions difficiles à créer à partir des instructions de base sont directement imprimées sur le silicium de la puce afin de gagner en rapidité d'exécution. L'inconvénient de ce type d'architecture provient justement du fait que des fonctions supplémentaires sont imprimées sur le silicium, d'où un coût élevé. D'autre part, les instructions sont de longueurs variables et peuvent parfois prendre plus d'un cycle d'horloge ce qui les rend lentes à l'exécution. Néanmoins, avec la considérable augmentation de la taille des puces électroniques et la gigantesque accélération des fréquences d'horloge, la puissance de calcul d'un microprocesseur CISC d'aujourd'hui est considérable. Son caractère généraliste lui permet d'être par excellence le composant de base de l'informatique. Mais en instrumentation et automatisme on lui préférera généralement des composants plus spécialisés ne nécessitant pas, ni un calcul complexe ni un traitement d'informations de masse. C'est pourquoi dans les applications industrielles, que ce soit d'automatisme ou d'instrumentation, le microcontrôleur est le composant programmable le plus utilisé. Il comporte sur sa puce un certain nombre d'interfaces qui n'existent pas sur un microprocesseur, par contre il est généralement moins puissant en terme de rapidité ou de taille de mémoire adressable et le plus souvent cantonné aux données de 8 ou 16 bits. Les microcontrôleurs utilisent la technologie RISC (Reduced Instruction Set Computer), dont la traduction est "ordinateur à jeu d'instructions réduit" et n'a pas de fonctions supplémentaires câblées. Ce qui qui implique une programmation plus difficile et un compilateur plus puissant. Les instructions d'un microcontrôleur sont tellement peu nombreuses (en moyenne une soixantaine) qu'il est possible de les graver directement sur le silicium sans alourdir de manière dramatique leur fabrication. L'avantage d'une telle architecture est bien évidemment le coût réduit au niveau de la fabrication des processeurs l'utilisant. De plus, les instructions, étant simples, ils sont exécutés en un cycle d'horloge, ce qui rend l'exécution des programmes plus rapides qu'avec des processeurs basés sur une architecture CISC. En plus, de tels processeurs sont capables de traiter plusieurs instructions simultanément en les traitant en parallèle. Les microcontrôleurs ont permis de faire évoluer les systèmes micro programmés vers encore plus de simplicité et de rapidité. Ils sont aujourd'hui utilisés dans la plupart des réalisations industrielles grand public ou professionnelles, ils gèrent au plus juste et au plus vite les applications. Leur évolution a permis l'intégration de circuits complexes variés. Ces

circuits ont été intégrés sur une même puce donnant ainsi beaucoup de flexibilité et de puissance de commande au microcontrôleur. Cette polyvalence lui permet d'occuper une place importante que ce soit en instrumentation, en commande ou en automatisme industriel. Le meilleur exemple est bien évidemment les automates programmables qui sont tous équipés de microcontrôleurs.

II. Description et structure interne d'un microcontrôleur.

Un microcontrôleur est un composant réunissant sur un seul et même silicium un micro-processeur, divers dispositifs d'entrées/sorties et de contrôle d'interruptions ainsi que de la mémoire, notamment pour stocker le programme d'application. Dédié au contrôle, il embarque également un certain nombre de périphériques spécifiques des domaines ciblés (bus série, interface parallèle, convertisseur analogique numérique, ...). Les microcontrôleurs améliorent l'intégration et le coût (lié à la conception et à la réalisation) d'un système à base de microprocesseur en rassemblant ces éléments essentiels dans un seul circuit intégré. On parle alors de "système sur une puce" (en anglais : "System On chip"). Il existe plusieurs familles de microcontrôleurs, se différenciant par la vitesse de leur processeur et par le nombre de périphériques qui les composent. Toutes ces familles ont un point commun c'est de réunir tous les éléments essentiels d'une structure à base de microprocesseur sur une même puce. Voici généralement ce que l'on trouve à l'intérieur d'un tel composant :

- Un microprocesseur (C.P.U.),
- Des bus,
- De la mémoire de donnée (RAM et EEPROM),
- De la mémoire programme (ROM, OTPROM, UVPROM ou EEPROM),
- Des interfaces parallèles pour la connexion des entrées / sorties,
- Des interfaces séries (synchrone ou asynchrone) pour le dialogue avec d'autres unités,
- Des timers pour générer ou mesurer des signaux avec une grande précision temporelle.

Sur la figure (2.1) nous présentons le schéma type d'un microcontrôleur.

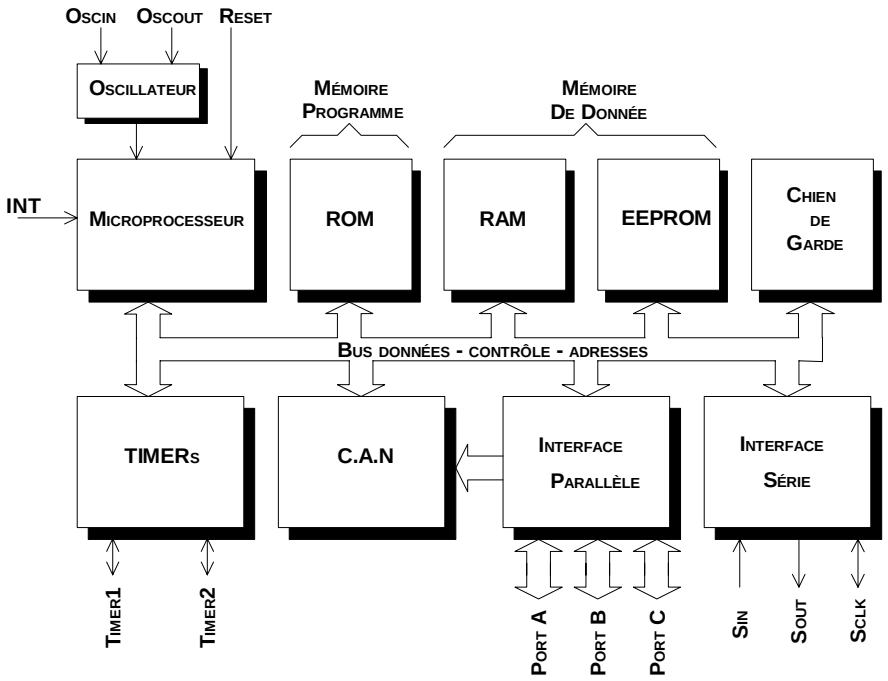


Figure 2.1. Schéma type de tout appareil programmable.

La présence de ces divers éléments de base est indispensable et, même s'ils ne sont pas toujours aussi visibles que sur notre synoptique, ils sont toujours présents. Ces éléments sont les mêmes que pour un système informatique classique mais, dans le cadre d'une application pouvant être traitée par un microcontrôleur, leurs tailles et fonctions différent un peu de ce que nous avons peut-être l'habitude de voir. L'unité centrale, généralement constituée par un microprocesseur plus ou moins évolué, exécute le programme qui va donner vie à l'application. Pour les applications industrielles, ces programmes ont en commun le fait qu'ils ne nécessitent que très rarement des calculs complexes alors qu'ils sont très friands de manipulations d'informations d'entrées/sorties. C'est pour cette raison que la majorité des microcontrôleurs inclut une interface parallèle de type PIA(Peripheral Interface Adapter figure (2.1)). Etant donné la nature des applications des microcontrôleurs souvent les programmes sont contenus dans le deuxième élément de la figure (2.1) qui est la mémoire morte ou ROM. Cette mémoire peut-être constituée de diverses

façons: mémoire programmée par masque, mémoire UVPROM ou EEPROM. De cette façon, nous n'avons plus besoin de charger le programme d'application à chaque mise sous tension. L'exécution de l'application est enclenchée automatiquement à la mise sous tension, ce qui n'est pas le cas pour les systèmes à base de microprocesseur qui nécessitent un programme de démarrage (système d'exploitation ou moniteur). Pour pouvoir travailler correctement notre microprocesseur a souvent besoin de stocker des données temporaires quelque part et c'est là qu'intervient la mémoire vive ou RAM qui, contrairement aux systèmes informatiques classiques, peut être de très petite taille. Le dernier élément qui ne manque pas d'importance dans les applications susceptible de faire appel à un microcontrôleur est tout ce qui concerne les circuits d'interface avec le monde extérieur. Contrairement aux systèmes informatiques classiques où les interfaces sont bien connues, de types assez peu diversifiés et en nombre relativement limité; les interfaces que nous pouvons rencontrer dans des applications industrielles sont absolument quelconques. Il y a en effet assez peu de points communs entre un moteur pas à pas, un afficheur à cristaux liquides ou bien encore un programmateur de machine à laver. Le schéma type d'un microcontrôleur est issu d'une analyse assez forte des divers systèmes de commande et d'automatisme réalisés avant l'avènement des microcontrôleurs. Les fabricants de circuits intégrés ont affinés un peu la définition de ce qu'il fallait intégrer pour arriver à un schéma-type analogue à celui de la figure (1.1). Nous y retrouvons bien évidemment un microprocesseur, généralement simplifié et avec un jeu d'instructions réduit mais auquel des instructions de manipulation de bits, très utiles pour faire des entrées/sorties, lui ont été ajoutées. Dans certains cas, ce microprocesseur se voit doté d'un très grand nombre de registres internes qui servent alors de mémoire vive. Ce qui évite l'utilisation d'une mémoire externe pour le stockage des données et rend le système encore plus compact. Dans un grand nombre de microcontrôleurs, nous trouvons également de la mémoire morte intégré. Avec les progrès technologiques les fabricants ont appris à placer sur la puce de la mémoire programmable électriquement et effaçable aux ultraviolets (UVPROM) ou, plus récemment encore, de la mémoire programmable et effaçable électriquement (EEPROM). On trouve donc à l'heure actuelle au moins quatre types différents de microcontrôleurs :

- ceux sans aucune mémoire morte ;

- ceux avec EPROM (programmable et effaçable à l'ultra violet);

-ceux avec EEPROM ;

-ceux avec un mélange de ces combinaisons (ROM -EEPROM ou UVPROM - EEPROM).

Pour ce qui est de la mémoire vive ou RAM, la situation est plus simple. Quasiment tous les microcontrôleurs disposent d'une RAM interne de taille en principe assez faible et, lorsqu'elle n'est pas explicitement visible sur le synoptique, c'est que l'unité centrale dispose d'assez de registres pour servir de RAM comme nous l'avons expliqué précédemment. Il est un peu plus délicat de faire un schéma type au niveau des circuits d'interface car c'est là que les différents microcontrôleurs se distinguent en fonction des créneaux d'applications qu'ils visent. Néanmoins, on rencontre généralement les éléments de base suivants :

- des lignes d'entrées/sorties parallèles en nombre variable selon la vocation et la taille du boîtier (un problème de nombre maximum de pattes se posant très vite avec l'accroissement du nombre de ces lignes);

- au moins une interface d'entrée/sortie série asynchrone, plus ou moins évoluée selon les circuits;

- un ou plusieurs timers internes dont les possibilités peuvent être très variables mais qui fonctionnent généralement en compteurs, décompteurs, générateurs d'impulsions programmables, etc. ;

- parfois, mais c'est un peu plus rare, un ou des convertisseurs analogiques numériques précédés ou non de multiplexeurs pour offrir plusieurs voies ;

- parfois aussi, mais c'est également plus rare, un convertisseur numérique analogique.

Enfin, bien que ce ne soit pas une véritable interface d'entrée/sortie au sens où nous l'entendons, certains microcontrôleurs disposent d'un accès à leur bus interne. Ceci permet de connecter des boîtiers destinés à accomplir des fonctions qui font défaut sur la puce ce qui est parfois utile. Précisons, mais c'est une évidence, que tous les microcontrôleurs sans mémoire morte interne disposent nécessairement de cette interface puisqu'il faut impérativement leur permettre d'accéder à une mémoire morte externe contenant le programme de l'application.

Bien sûr, notre schéma-type ne peut recouvrir tous les cas mais il vous donne une idée assez précise de ce que contient un microcontrôleur. Il est bien évident d'après ce schéma que la puissance d'un microcontrôleur est directement liée au processeur qu'il intègre. C'est pourquoi les constructeurs développent souvent un cœur de processeur destiné aussi bien à décliner une gamme de microprocesseurs que de microcontrôleurs ; La tendance aujourd'hui et un cœur de processeur 16 ou 32 bits qui représente une augmentation de la surface occupée sur le silicium de seulement quelques pour-cent par rapport à un circuit en 8 bits. Le prix du microcontrôleur étant directement lié à sa taille, embarquer une puissance supérieure est un choix qui ne grève plus le budget. Opter pour un cœur de processeur en 16 ou en 32 bits, c'est permettre entre autres des exécutions parallèles, un espace mémoire élargi, des interfaces de communications ou encore le remplacement de fonctions analogiques par des traitements numériques. Le cœur de processeur 32 bits le plus prisé aujourd'hui est l'ARM7. Pour preuve, plusieurs fabricants l'utilisent aujourd'hui pour le développement de microcontrôleurs 32 bits. Les secteurs les plus visés sont surtout les systèmes de communication sans fil, les téléphones portables et l'industrie automobile. Par exemple les microcontrôleurs de la famille STR7 de STMicroelectronics sont basés sur le cœur ARM7. Ces microcontrôleurs intègrent une multitude d'interfaces série (CAN, USB, I2C, SPI,..), de la mémoire SDRAM, de la mémoire FLASH, des convertisseurs analogiques numériques, 48 entrées/sorties, une interface JTAG, plusieurs horloges temps réel,.. L étude d'un microcontrôleur basé sur le cœur ARM7 comme le STR7 dépasse largement le cadre de ce cours. La diversité des périphériques qu'il intègre rend sa compréhension très difficile c'est pourquoi nous préférons présenter un cœur de processeur beaucoup plus simple comme exemple d'étude dans le cadre de ce cours. Dans le paragraphe suivant nous allons présenter les éléments généraux communs aux divers types de processeurs.

III. Le processeur.

Le processeur (microprocesseur) est le composant hardware le plus connu d'un système micro-programmé. C'est l'unité intelligente de traitement des informations. Son travail consiste à lire des programmes (des suites d'instructions), à les décoder et à les exécuter. Les années 80 voyaient l'émergence de ces circuits avec les Zylog Z80, 6800 de Motorola, le 8085 de Intel qui est souvent utilisé en tant que microcontrôleur. Avec l'arrivée des PC-XT d'IBM et l'utilisation du 8088, INTEL devenait maître du marché fin des années 80. L'interfaçage du processeur avec l'extérieur nécessite 3 bus: un bus de données, un bus d'adresse et un bus de commande. Il existe des processeurs basés sur l'architecture CISC et d'autres basés sur l'architecture RISC. Cependant certains processeurs sont difficilement classifiables comme le

CPU i486 également appelé 80486. Ce véritable processeur 32 bits (bus internes et externes) est à mi chemin entre le tout CISC et le tout RISC. Ce processeur offre des performances similaires aux processeurs RISC 32 bits, tout en restant compatible 100% avec son prédécesseur CISC le 80386. Par contre le processeur 68040 de Motorola est basé sur l'architecture CISC. Il est présenté par Motorola comme étant un processeur CISC pur et dur. Les premiers concepteurs de processeurs rajoutaient le plus d'instructions possibles pour permettre à l'utilisateur de peaufiner ses programmes. Néanmoins, ces multiples instructions ralentissent le fonctionnement du microprocesseur et sont peu utilisées en pratique. Actuellement, on utilise de plus en plus de processeurs **RISC** (Reduced Instruction Set Computer). Le nombre d'instructions est réduit, mais exécutées nettement plus rapidement. Chaque instruction complexe peut être programmée par plusieurs instructions simples. Un processeur est constitué de:

- une unité de commande qui lit les instructions et les décode;
- une unité de traitement (UAL - unité arithmétique et logique) qui exécute les instructions;
- d'un ensemble de mémoire appelés registres;
- d'un bus de données externe;
- d'un bus d'adresse externe;
- d'un bus de commande externe;
- d'un bus de données interne reliant l'unité de commande l'UAL et les registres.

Lorsque tous ces éléments sont regroupés sur une même puce, on parle alors de microprocesseur. La figure ci dessous donne une idée sur l'architecture interne d'un microprocesseur. Sur cette figure nous pouvons voir les 3 bus qui permettent au microprocesseur de communiquer avec l'extérieur.

III.1 Architecture de base d'un microprocesseur

III.1.1 L'unité de commande.

Elle permet de "séquencer" le déroulement des instructions. Elle effectue la recherche en mémoire de l'instruction, le décodage, l'exécution et la préparation de l'instruction suivante. L'unité de commande élabore tous les signaux de synchronisation internes ou externes (bus des commandes) au microprocesseur.

III.1.2 L'unité arithmétique et logique (UAL).

C'est l'organe qui effectue les opérations:

Arithmétiques : addition, soustraction, multiplication, ...

Logiques : et, ou, non, décalage, rotation, ...

Deux registres sont associés à l'UAL : l'accumulateur et le registre d'état.

III.1.2.1 L'accumulateur (nommé : A).

C'est une des deux entrées de l'UAL. Il est impliqué dans presque toutes les opérations réalisées par l'UAL. Certains constructeurs ont des microprocesseurs à deux accumulateurs (Motorola : 6800).

Exemple: **A** étant l'accumulateur et **B** un registre, on peut avoir : A+B (ADD A,B : addition du contenu du registre A avec celui du registre B, le résultat étant mis dans A)

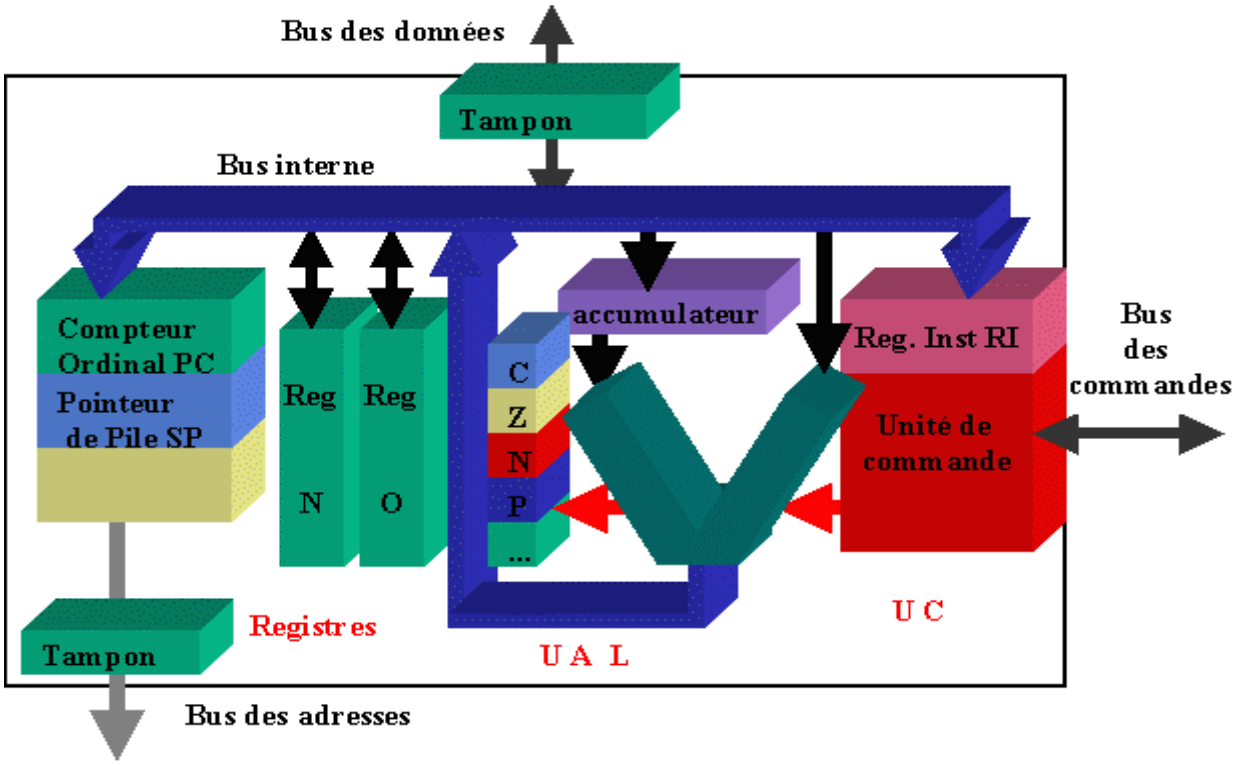


Figure 2.2. Architecture de base de microprocesseur.

III.1.2.2 Le registre d'état (Flags : F)

A chaque opération, le microprocesseur positionne un certain nombre de bascules d'état. Ces bascules sont appelées aussi indicateurs d'état ou drapeaux (status, flags). Par exemple, si une soustraction donne un résultat nul, l'indicateur de zéro (Z) sera mis à **1**. Ces bascules sont regroupées dans le registre d'état

On peut citer comme indicateurs:

- retenue (carry : C)
- retenue intermédiaire (Auxiliary-Carry : AC)
- signe (Sign : S)
- débordement (Overflow : O)
- zéro (Z)
- parité (Parity : P)

III.1.2.2.1 Retenue : (carry : C).

Exemple: addition de nombres binaire sur 8 bits

		11111100			**FCH**
	+	**10000010**		+	**82H**
carry : **1**	=	**01111110**	carry : **1**	=	**7EH**

Sur cet exemple, nous pouvons remarquer que le résultat sur 8 bits est faux (7EH). Toutefois le bit de retenu expulsé n'est pas définitivement perdu. Ce bit C (carry) est stocké dans le registre d'état est peut être testé par l'utilisateur. De la même manière lors d'une opération de décalage ou de rotation ce bit peut être positionné en cas de débordement. Par exemple, soit le décalage à gauche sur d'un bit de cet octet : 10010110, après le décalage nous avons 00101100. En ce qui concerne le carry, il va être positionné à **1**, puisque le bit expulsé est égal à **1**.

III.1.2.2.2 Retenue intermédiaire : (Auxiliary Carry : AC).

Sur les opérations arithmétiques, ce bit signale une retenue entre groupes de 4 bits (Half-byte: demi-octet) d'une quantité de 8 bits.

III.1.2.2.3 Signe: (S)

Ce bit est mise à **1** lorsque le résultat de l'opération est négatif (MSB: bit de plus fort poids du résultat: à **1**).

III.1.2.2.4 Débordement : (overflow : O)

Cet indicateur est mis **1**, lorsqu'il y a un dépassement de capacité pour les opérations arithmétiques en complément à 2. Sur 8 bits, on peut coder de -128 (1000 0000) à +127 (0111 1111).

104		**0110 1000**	**- 18**	**1110 1110**	
+ **26**	**+**	**0001 1010**	**- 118**	**1000 1010**	
=130	**=**	**1000 0010 (-126)**	**-136**	**0111 1000 (120) avec C=1**	

Dans cet exemple et dans les deux cas de figure (opération à droite ou opération à gauche), le bit O est positionné à **1**. L'indicateur de débordement est une fonction logique (OU exclusif) de la retenue (C) et du signe (S).

III.1.2.2.5 Le bit Zéro : (Zéro : Z)

Ce bit est mis à **1** lorsque le résultat de l'opération est nul.

III.1.2.2.6 Le bit de parité : (P)

Ce bit est mis à 1 lorsque le nombre de 1 de l'accumulateur est pair.

Remarque : La plupart des instructions modifient le **registre d'état.**

Exemple :

ADD A, B positionne les drapeaux : **O, S, Z, A, P, C**

OR B, C (B ou C -> B) positionne **S, Z, P** tandis que

MOV A, B (Move, Transférer le contenu de **B** dans **A**) n'en positionne aucun.

III.1.3 Les registres.

Il y'a deux type de registres : les registres d'usage général, et les registres d'adresses (pointeurs).

III.1.3.1 Les registres d'usage général.

Ce sont des mémoires rapides, à l'intérieur du microprocesseur, qui permettent à l'UAL de manipuler des données à vitesse élevée. Ils sont connectés au bus de données interne au microprocesseur. L'adresse d'un registre est associée à son nom (on donne généralement comme nom une lettre) A, B,C...

Exemple : **MOV C,B**: transfert du contenu du registre **B** dans le registre **C**.

III.1.3.2 Les registres d'adresses (pointeurs).

Ce sont des registres connectés sur le bus d'adresses. On peut citer comme registre:

- Le compteur ordinal (pointeur de programme PC) ;
- Le pointeur de pile (stack pointer SP) ;
- Les registres d'index (index source SI et index destination DI).

III.1.3.2.1 Le compteur ordinal (pointeur de programme PC.)

Il contient l'adresse de l'instruction à rechercher en mémoire. L'unité de commande incrémente le compteur ordinal (PC) du nombre d'octets sur lequel l'instruction, en cours d'exécution, est codée. Le compteur ordinal contiendra alors l'adresse de l'instruction suivante. Prenons l'exemple d'un microprocesseur 8086 de INTEL :

Exemple : (**PC**)=**10000H** ; il pointe la mémoire qui contient l'instruction **MOV CX,BX** qui est codée sur deux octets (**89 D9H**) ; l'unité de commande incrémentera de deux le contenu du

PC : (**PC**) = **10002H** (la mémoire sera supposée être organisée en octets).

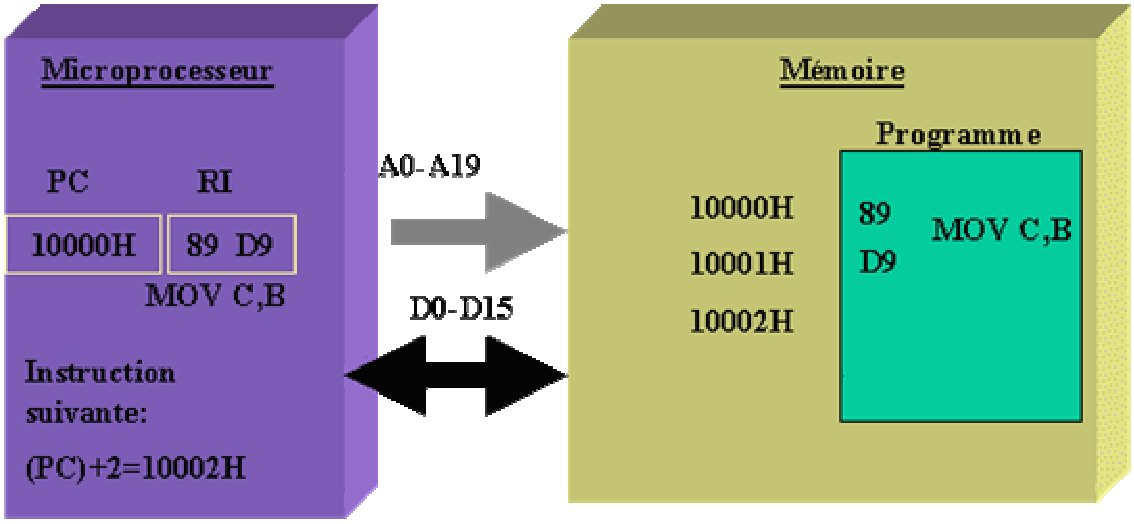


Figure 2.3 Compteur de Programme (PC).

III.1.3.2.2 Le pointeur de pile (stack pointer SP).

Il contient l'adresse de la pile. Celle-ci est une partie de la mémoire, elle permet de stocker des informations (le contenu des registres) relatives au traitement des interruptions et des sous-programmes. La pile est gérée en **LIFO** : (Last IN First Out) dernier entré premier sorti. Le fonctionnement est identique à une pile d'assiettes. Le pointeur de pile **SP** pointe le haut de la pile (**31000H** figure 2.4), il est décrémenté avant chaque empilement, et incrémenté après chaque dépilement. Il existe deux instructions pour empiler et dépiler: **PUSH** et **POP**. exemple: **PUSH A** empilera le registre A et **POP A** le dépilera. La figure suivante montre l'évolution du pointeur de pile durant l'exécution du programme commençant en **12E30H.**

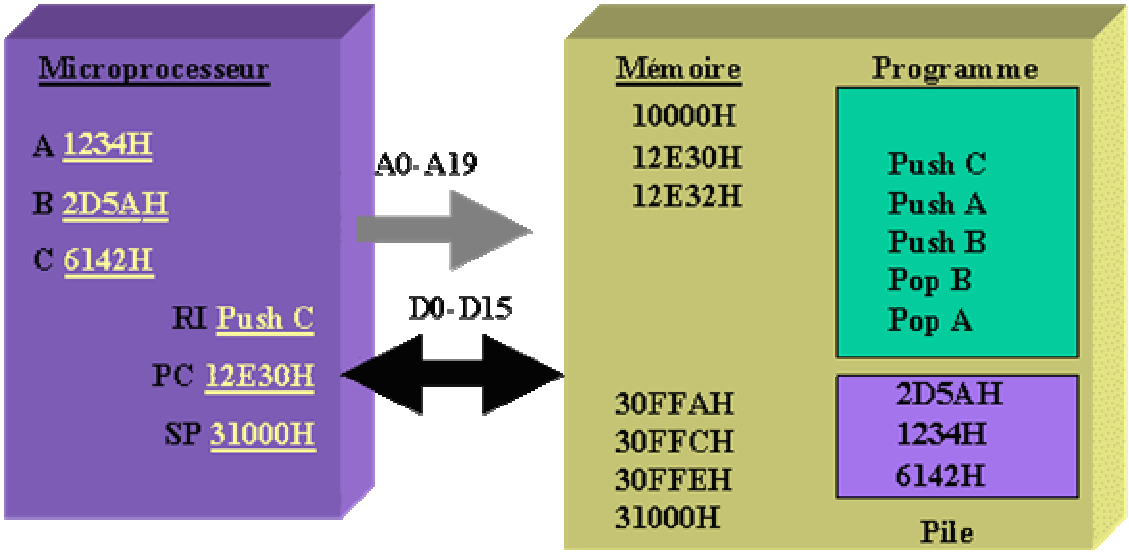


Figure 2.4. Principe de fonctionnement de la pile.

Sur cette figure le programme commence par sauvegarder le contenu de **C** dans la pile (**PUSH C**). Pour cela (**SP**) est décrémenté de deux ((**SP**)=**31000H-2=30FFEH**), puis on effectue l'écriture de (**C**) dans la mémoire à l'adresse (**SP**) : (**30FFEH**) = **6142H**. Pour **PUSH A** on obtient : (**30FFCH**)=**1234H**, et pour **PUSH B** : (**30FFAH**)=**2D5AH**. Pour l'instruction **POP B**, (**SP**) est chargé dans le registre B (**SP**=**30FFAH** ; **B**=**2D5AH**) puis (**SP**) est incrémenté de deux (**SP**= **30FFAH+2**=**30FFCH**). Enfin, pour **POP A** on obtient : **A**=**1234H** et (**SP**=**30FFCH + 2** = **30FFEH).**

III.1.3.2.3 Les registres d'index (index source SI et index destination DI).

Les registres d'index permettent de mémoriser une adresse particulière (par exemple : début d'un tableau). Ces registres sont aussi utilisés pour adresser la mémoire de manière différente. C'est le mode d'adressage indexé.

Exemple :

MOV A,[SI+10000H] place le contenu de la mémoire d'adresse **10000H** + le contenu de **SI**, dans le registre **A**.

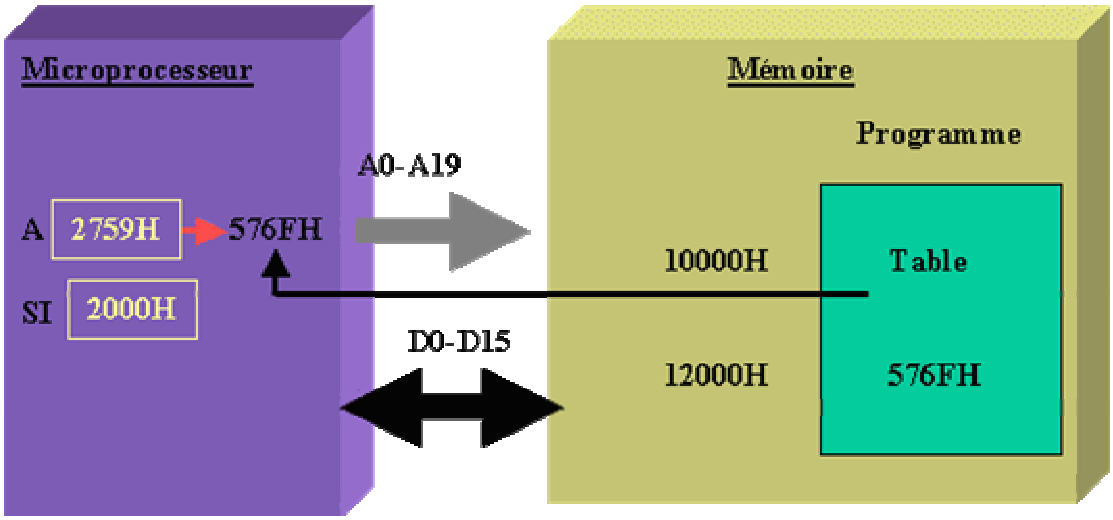


Figure 2.5. Exemple d'opération sur le registre d'index.

III.2 Principe d'exécution d'une instruction.

Dans cette architecture standard, l'exécution d'une instruction se fait en trois étapes:

-Recherche de l'instruction **(Fetch) ;**

-Décodage (**decode**) ;

-Exécution (**execute**).

III.2.1 Recherche de l'instruction.

Le contenu de PC (compteur ordinal) est placé sur le bus d'adresse (c'est l'unité de commande qui établit la connexion). L'unité de commande (UC) émet un ordre de lecture (READ=RD=1). Au bout d'un certain temps (temps d'accès à la mémoire), le contenu de la case mémoire sélectionnée est disponible sur le bus des données. L'unité de commande charge la donnée dans le registre d'instruction pour décodage. Le microprocesseur place le contenu de PC (**10000H**) sur le bus d'adresse et met RD à 1 (cycle de lecture). La mémoire met sur le bus de données le contenu de sa mémoire n° **10000H** (ici **89D9H** qui est le code de **MOV C,B**). Le microprocesseur place dans son registre d'instruction le contenu du bus de données (**89D9H**). L'unité de commande décode et exécute l'instruction **MOV C,B**.

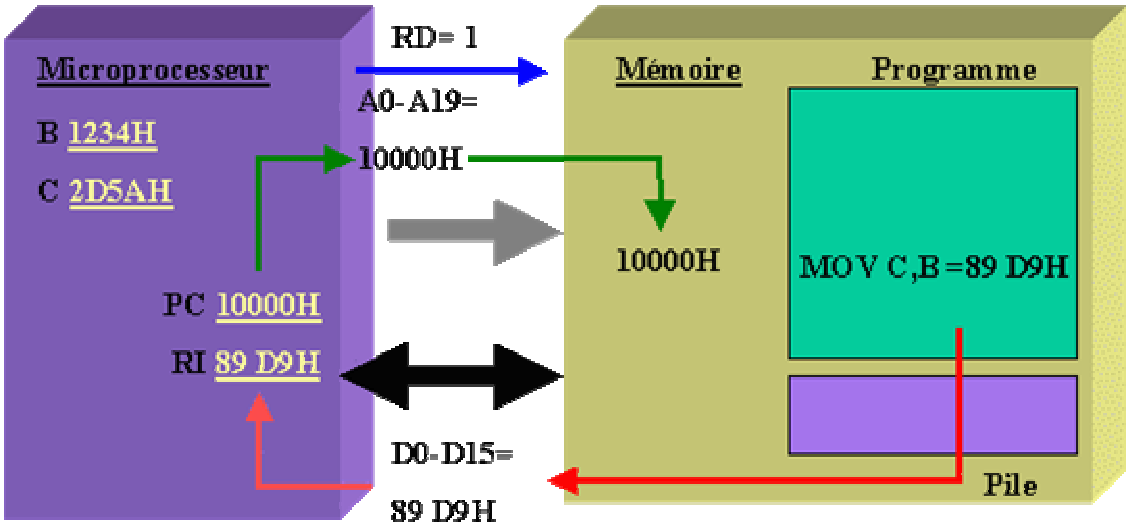


Figure 2.6 Compteur de Programme (PC).

III.2.2 Le Décodage de l'instruction.

Le registre d'instruction contient maintenant le premier mot de l'instruction qui peut être codée sur plusieurs mots. Ce premier mot contient le **code opératoire** qui définit la nature de l'opération à effectuer (addition, rotation,...) et le nombre de mots de l'instruction. L'unité de commande décode le code opératoire et peut alors exécuter l'instruction.

III.2.3 L'exécution de l'instruction.

Après l'exécution de l'instruction par micro-programme, les indicateurs sont positionnés (**O, S, Z, A, P, C**). L'unité de commande positionne le compteur ordinal (PC) pour l'instruction suivante.

IV. Les BUS.

Comme nous l'avions vu plus haut, les trois éléments fondamentaux d'un système micro-programmé sont : le microprocesseur ou microcontrôleur, la mémoire et les boîtiers d'entrées sorties. Tous ces éléments sont reliés entre eux par des bus comme le montre la figure ci dessous :

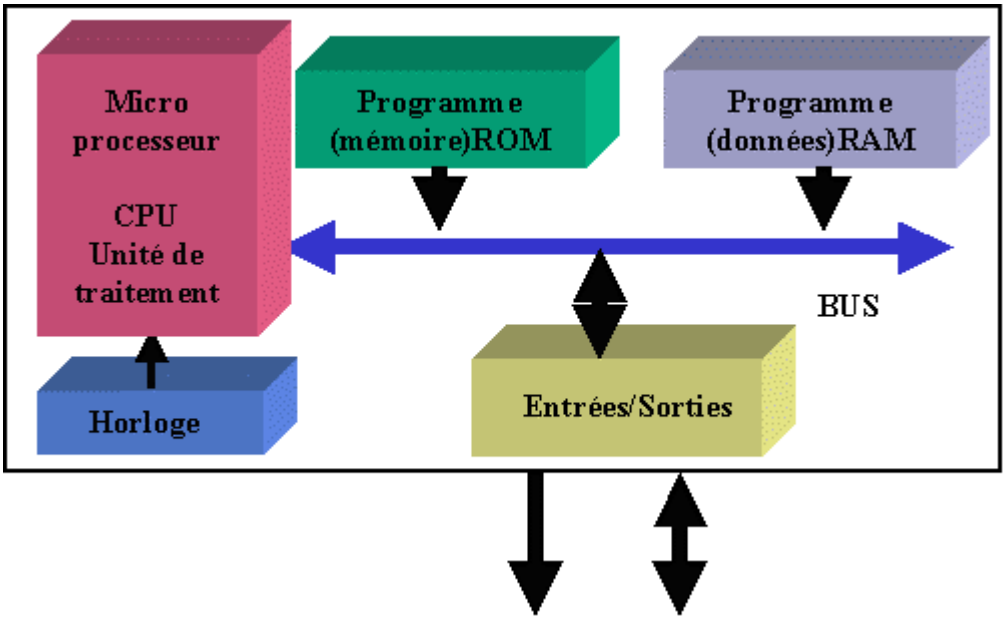


Figure 2.7. Architecture de base d'un micro-ordinateur

On appelle Bus, en informatique, un ensemble de liaisons physiques (câbles, pistes de circuits imprimés, ...) pouvant être exploitées en commun par plusieurs éléments matériels afin de communiquer. Les Bus ont pour but de réduire le nombre de traces nécessaires à la communication des différents composants en mutualisant les communications sur une seule voie de données. Dans le cas où la ligne sert uniquement à la communication de deux composants matériels, on parle parfois de port (*port série*, *port parallèle*, ...). Un Bus est caractérisé par le volume d'informations transmises simultanément (exprimé en bits), correspondant au nombre de lignes sur lesquelles les données sont envoyées de manière simultané. Une nappe de 32 fils permet ainsi de transmettre 32 bits en parallèle. On parle ainsi de "largeur de bus" pour désigner le nombre de bits qu'il peut transmettre simultanément. D'autre part, la vitesse du bus est également définie par sa fréquence (exprimée en Hertz), c'est-à-dire le nombre de paquets de données envoyés ou reçus par seconde. On parle de **cycle** pour désigner chaque envoi ou réception de données. De cette façon, il est possible de connaître la bande passante d'un bus, c'est-à-dire le débit de données qu'il peut transporter, en multipliant sa largeur par sa fréquence. Un Bus d'une largeur de 16 bits, cadencé à une fréquence de 133 Mhz possède donc une bande passante égale à :

$16 * 133.10^6 = 2128*10^6$ bit/s,

soit $2128*10^6/8 = 266*10^6$ octets/s

soit $266*10^6 /1024 = 259.7*10^3$ Ko/s

soit $259.7*10^3$ /1024 = 253.7 Mo/s

Pour la communication, un microprocesseur a besoin en général de trois Bus. Un Bus de données, un Bus d'adresse et un Bus de commande. Sur la figure suivante nous présentons un système à base de microprocesseur classique avec ces trois Bus. Chaque Bus a une fonction particulière :

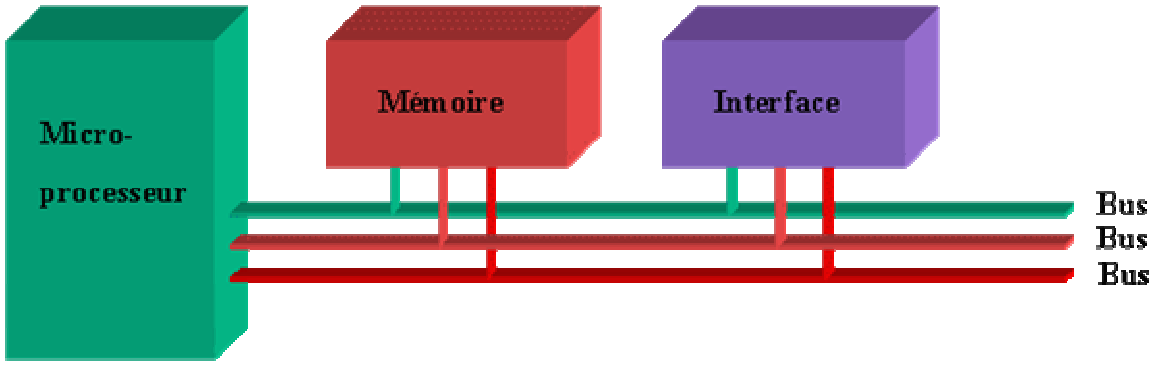


Figure 2.8. Les bus d'extensions

IV.1 Bus de données.

Il permet de véhiculer des données du microprocesseur vers un composant ou d'un composant vers le microprocesseur. Il est donc bidirectionnel. Le nombre de fils de ce bus varie suivant les microprocesseurs (8 / 16 / 32 / 64 bits). Dans la littérature, les différents fils de ce bus sont appelés D0, D1, ..., Dp-1, si le bus a "p" fils.

IV.2 Bus d'adresse.

La mémoire est composée de nombreuses cases mémoires. Chaque case est repérée par une adresse. Lorsque le microprocesseur veut, par exemple, lire une case, il doit indiquer à quelle adresse elle se trouve. Il met cette adresse sur le bus des adresses. La case mémoire reconnaît alors son adresse et met sur le bus de données son contenu.

Exemple : Bus d'adresse 16 bits - données sur 8 bits.

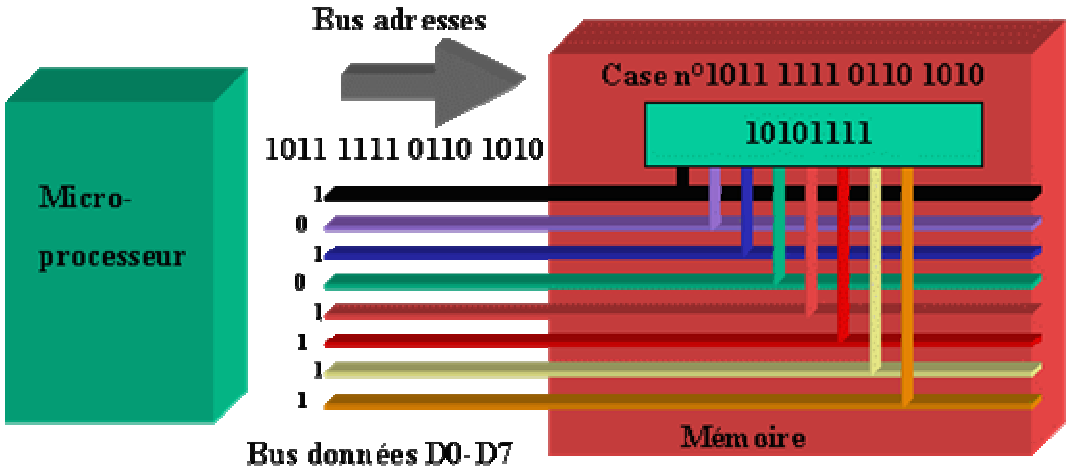


Figure 2.9. Adressage de la mémoire.

Dans l'exemple précédent, le microprocesseur écrit la donnée 10101111 dans la case mémoire d'adresse 1011 1111 0110 1010.

Le bus d'adresse est unidirectionnel : du microprocesseur vers les autres composants. Il se compose de 16 à 32 fils, suivant les microprocesseurs, que l'on nomme A0, A1, ..., An-1. Le tableau suivant donne l'espace mémoire adressable en fonction du nombre des lignes d'adresses.

16 bits		adressage de 2^{16}		64x1024 mots = 64 Kmots
20 bits		adressage de 2^{20}		1024x1024 mots = 1Mmots
32 bits		adressage de 2^{32}		4096x1024 x1024 mots = 4 Gmots

Tableau 1. Espace mémoire adressable en fonction des lignes d'adresses.

IV.3 Bus des commandes.

Le bus des commandes est constitué d'un ensemble de fils de "commandes", permettant la synchronisation et bien sûr la commande des boîtiers mémoires et entrées/sorties par le microprocesseur. Dans le cas précédent, la cellule mémoire doit savoir à quel instant elle doit mettre son contenu sur le bus de données. Pour cela, le microprocesseur possède une broche appelée Read ($\overline{RD}$) qu'il met à 0 (0v) lorsque la cellule doit agir. De même, lors d'une écriture du microprocesseur vers la

cellule, il met sa broche Write ($\overline{WR}$) à 0 (0V). Les signaux RD et WR sont des signaux de synchronisation, de contrôle et de commande. Ils sont reliés aux autres composants par un bus: le bus des commandes. Celui-ci comporte d'autres signaux de commandes.

V. Les Mémoires.

La mémoire vive, généralement appelée RAM (*Random Access Memory*, *mémoire à accès aléatoire*), est la mémoire principale du système, c'est-à-dire qu'il s'agit d'un espace permettant de stocker de manière temporaire des données lors de l'exécution d'un programme. En effet le stockage de données dans la mémoire vive est temporaire, contrairement au stockage de données sur une mémoire de masse telle que le disque dur, car elle permet uniquement de stocker des données tant qu'elle est alimentée électriquement. Ainsi, à chaque fois que l'ordinateur est éteint, toutes les données présentes en mémoire sont irrémédiablement effacées.

La mémoire morte, appelée ROM pour *Read Only Memory* (*mémoire à lecture seulement*) est un type de mémoire permettant de conserver les informations qui y sont contenues même lorsque la mémoire n'est plus alimentée électriquement. A la base ce type de mémoire ne peut être accédée qu'en lecture. Toutefois il est désormais possible d'enregistrer des informations dans certaines mémoires de type *ROM*.

V.1 Fonctionnement de la mémoire vive.

La mémoire vive est constituée de centaines de milliers de petits condensateurs emmagasinant des charges. Lorsqu'il est chargé, l'état logique du condensateur est égal à 1, dans le cas contraire il est à 0, ce qui signifie que chaque condensateur représente un bit de la mémoire. Etant donné que les condensateurs se déchargent, il faut constamment les recharger (le terme exact est *rafraîchir*) à un intervalle de temps régulier appelé **cycle de rafraîchissement** (d'une durée d'environ 15 nanosecondes (ns) pour une mémoire DRAM). Chaque condensateur est couplé à un transistor (de type *MOS*) permettant de "récupérer" ou de modifier l'état du condensateur. Ces transistors sont rangés sous forme de tableau (matrice), c'est-à-dire que l'on accède à une "case mémoire" (aussi appelée *point mémoire*) par une ligne et une colonne. Un boîtier mémoire est donc constitué d'un ensemble d'entités mémoire élémentaires (cellules mémoire) stockant un élément binaire (bit : **B**inary dig**IT)** ayant pour valeur 0 ou 1**.** Ces cellules sont groupées en mot **(word) de p bits (en général p=1 ou 8 bits**). Le nombre n de

cases mémoire de p bits appelé capacité ou taille de la mémoire s'exprime en Kilo **(1Ko=2^{10}=1024)** ou en Méga (1Mo=2^{20}=1024*1024=1048576).

Chaque point mémoire est donc caractérisé par une adresse, correspondant à un numéro de ligne et un numéro de colonne. Or cet accès n'est pas instantané et s'effectue pendant un délai appelé **temps de latence**. Par conséquent l'accès à une donnée en mémoire dure un temps égal au temps de cycle auquel il faut ajouter le temps de latence. Ainsi, pour une mémoire de type DRAM, le temps d'accès est **de 60 nanosecondes** (**35 ns** de délai de cycle et **25 ns** de temps

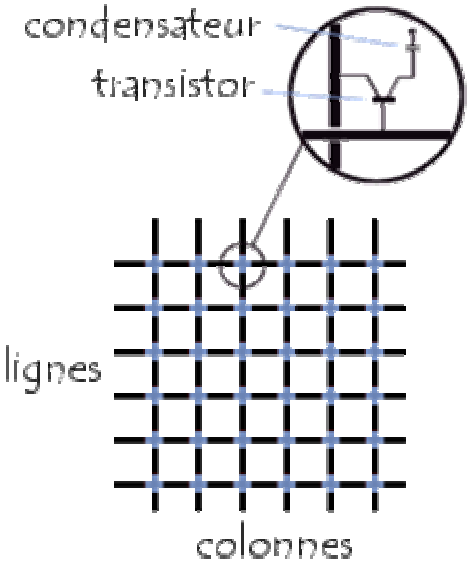


Figure 2.9. Schéma interne d'une mémoire vive.

de latence). Sur un ordinateur, le temps de cycle correspond à l'inverse de la fréquence de l'horloge, par exemple pour un ordinateur cadencé à 200 Mhz, le temps de cycle est de 5ns ($1/(200.10^6)$). Par conséquent un ordinateur ayant une fréquence élevée et utilisant des mémoires dont le temps d'accès est beaucoup plus long que le temps de cycle du processeur doit effectuer des **cycles d'attente** (en anglais ***wait state*)** pour accéder à la mémoire. Dans le cas d'un ordinateur cadencé à 200Mhz utilisant des mémoires de types DRAM (dont le temps d'accès est de 60ns), il y a **11 cycles d'attente** pour un cycle de transfert. Les performances de l'ordinateur sont d'autant diminuées qu'il y a de cycles d'attentes, il est donc conseillé d'utiliser des mémoires plus rapides.

Les mémoires sont connectées à un bus d'adresse de n bits, un bus de données de p bits et des lignes de commandes (figure 2.8). Pour pouvoir communiquer avec le microprocesseur, on va

relier leurs bus ensembles. Pour cela, il est nécessaire d'avoir adéquation entre le nombre de bits des bus de données et d'adresse de la mémoire et du microprocesseur.

V.2 Sélection d'une case mémoire.

Sur la figure 2.10 apparaît une broche de validation. Elle permet de sélectionner un boîtier mémoire parmi plusieurs, d'où son appellation : "chip select". Cette broche permet d'éviter les conflits sur le bus de données. En effet dans le cas général, il existe plusieurs boîtiers mémoire sur la carte, tous branchés sur le même bus de données. Dans ce cas, il est nécessaire de construire un signal qui permettra à un seul boîtier d'accéder au bus de données. Ce signal

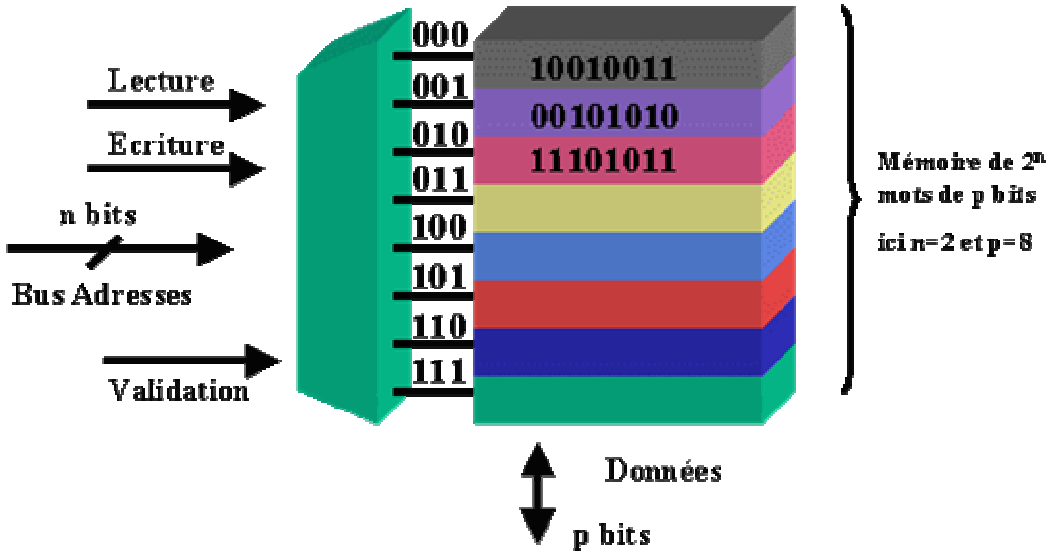


Figure 2.10. Organisation externe de la mémoire.

est appelé CS (chip select) sélection de boîtier ou CE (chip enable) validation de boîtier. Il faut créer autant de CS qu'il y a de boîtiers. Dans notre exemple figure 2.11, il nous faut fabriquer deux CS : CS1, CS2

Exemple :

Le bus d'adresse est sur 16 bits, le bus de données est sur 8 bits (figure 2.11). A l'adresse 1000H, le premier boîtier mémoire contient A7H et le second contient A6H. Si le microprocesseur fait une lecture à l'adresse 1000H (RD=1). Le premier boîtier mettra **A7**H sur le bus de données et le deuxième **A6H** sur le bus de données. Le bus de données D7- D0 a donc sur son fil D0 un "**0**" et un "**1**" ; c'est à dire 5 volts et la masse. C'est donc un **court-circuit.**

Les sélections de boîtiers "CS" (CS1 et CS2 sur la figure suivante) sont des fonctions logiques. Elles proviennent de circuits combinatoires appelés "logique de décodage ou encore décodage adresse". Les variables logiques de ces fonctions logiques sont

les variables du bus d'adresse (A0-An-1). Le choix des plages de validation des CSi sont exclusives les unes par rapport aux autres. C'est à dire qu'elles ne se recouvrent pas.

Si CSi(A0-An-1) = 1 =>CSj(A0-An-1) = 0 quelque soit $j \neq i$

Exemple:

Prenons l'exemple de la figure 2.13, si le bus d'adresse se compose de 16 fils (A0-A15). Supposons que la taille mémoire des deux boîtiers soit 32*1024 = 32K = 2^{15} adresses.

Il y a donc 15 broches adresses sur chaque boîtier. Nous pourrons mettre les fils A0-A14 du bus d'adresse sur ces broches. On peut prendre CS1 = /A15 (complément de A15) et

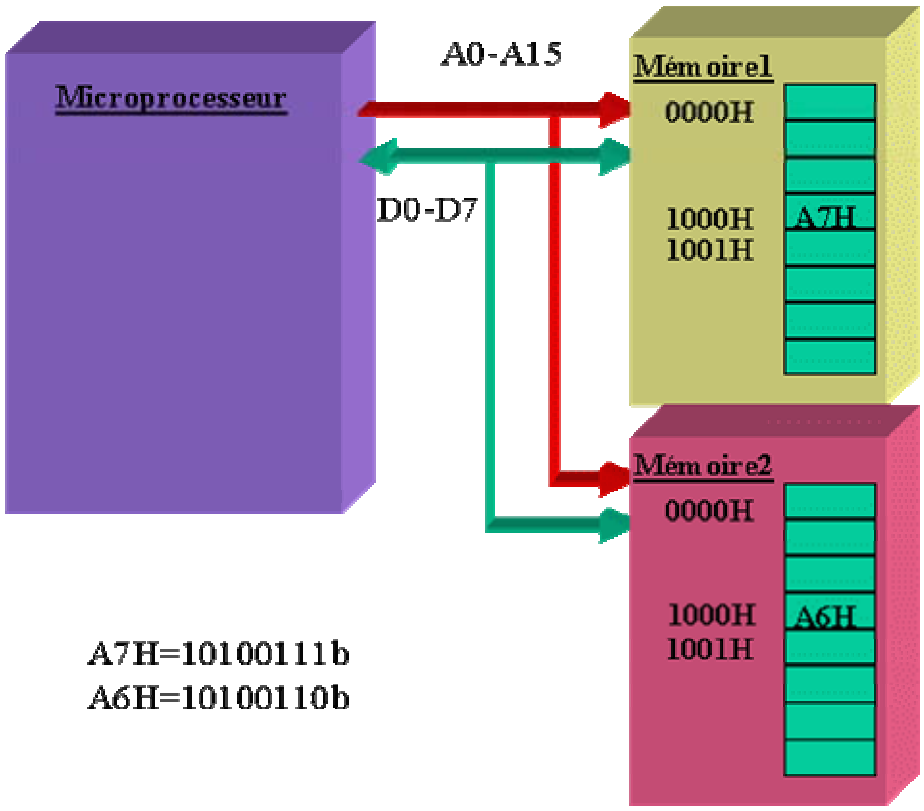


Figure 2.11 Conflit sur un bus.

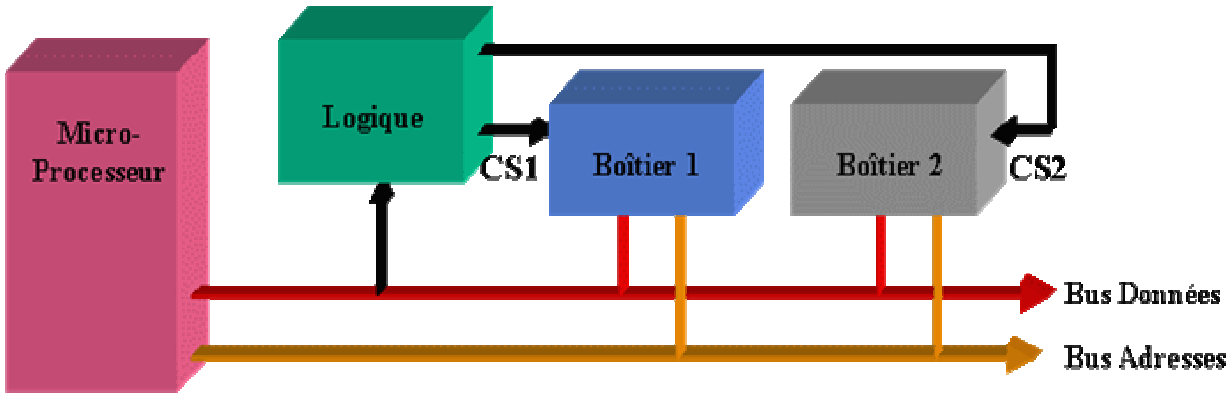


Figure 2.12 Sélection de la mémoire.

CS2=A15. Une lecture du microprocesseur à l'adresse **1000H** donnera : CS1 =1 et CS2 =0. Ce sera donc le premier boîtier qui sera validé et qui mettra le contenu de sa mémoire n° 0001 0000 0000 0000 sur le bus de données **(A7H** si on reprend l'exercice précédent). L'assemblage de plusieurs boîtiers forme un plan mémoire de plus grande capacité. L'assemblage horizontal (en largeur) permet de réaliser des mémoires de mots plus grands. Les boîtiers partagent le même bus d'adresses et de contrôle. Tandis que l'assemblage vertical (en profondeur) augmente la capacité (taille) mémoire du micro-ordinateur, les boîtiers partagent le même bus de données.

Exemple :

Nous disposons d'un microprocesseur utilisant un bus de données de dimension 16 fils (D0-D15) et pouvant adresser 1 Mo cases de mémoire (bus d'adresse sur 20 fils A0-A19 : 2^{20}=1Mo). Nous disposons également de boîtiers mémoire de 128 Ko octets chacun. De plus, nous désirons travailler sur des mots de 16 bits. Chaque boîtier mémoire a besoin de17 fils du bus d'adresse A1-A17 (2^{17}=128K) pour être branchés sur ses broches d'adresse. Il reste donc A0, A18 et A19. On va se servir de A18 et A19 pour construire les quatre CS des 8 boîtiers mémoire et de A0 pour sélectionner partie basse ou haute du bus de données. La solution de ce problème se trouve dans la figure (2.15), où nous utilisons un décodeur d'adresse à deux entrées de sélection et quatre sorties (CSi). On a immédiatement l'expression des fonctions logiques CS0, CS1, CS2, CS3. (CS0=/A19*/A18 CS1=/A19*A18 CS2=A19*/A18 CS3=A19*A18)

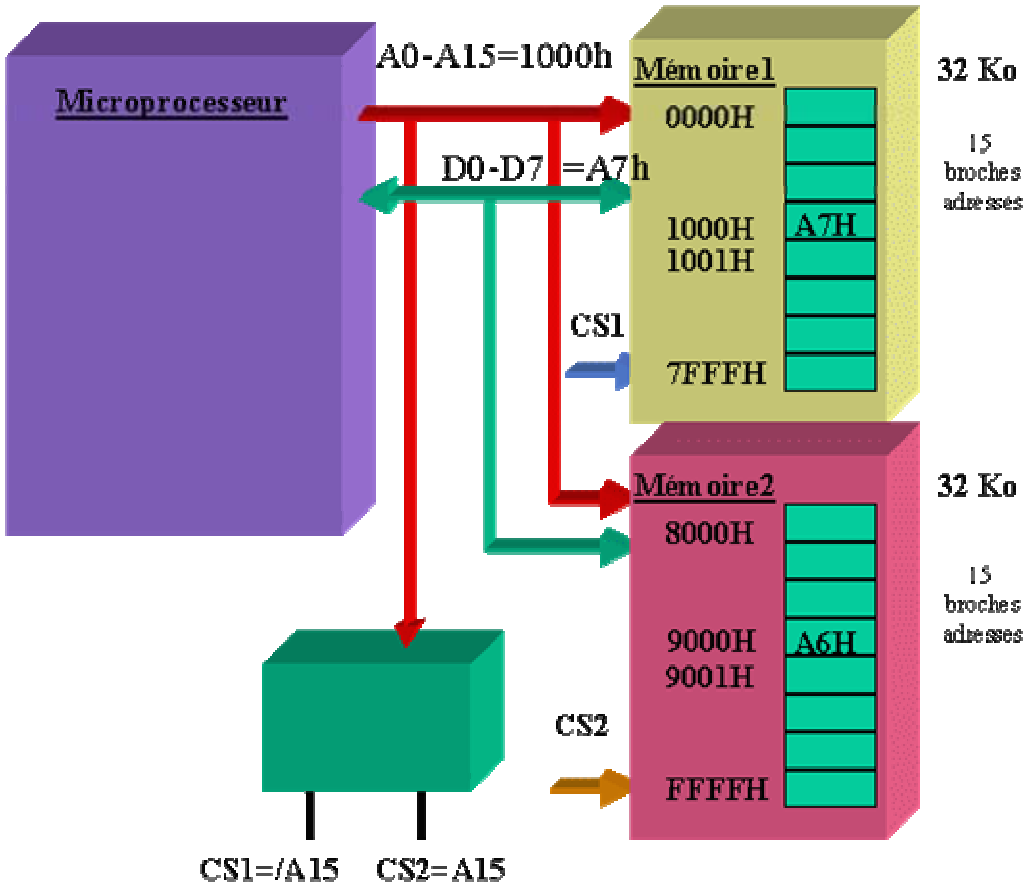
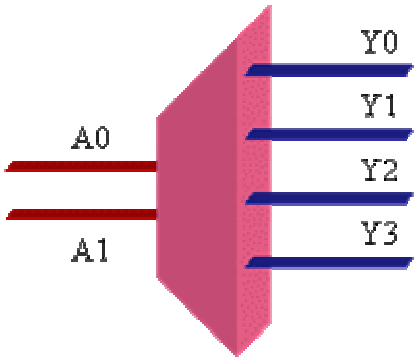


Figure 2.13 Réalisation d'une sélection boîtier.

V.3 Rappel sur les décodeurs.



A1	A0	Y3	Y2	Y1	Y0	
0	0	0	0	0	1	
0	1	0	0	1	0	
1	0	0	1	0	0	
1	1	1	0	0	0	

Figure 2.14. Le décodeur 2 -> 4.

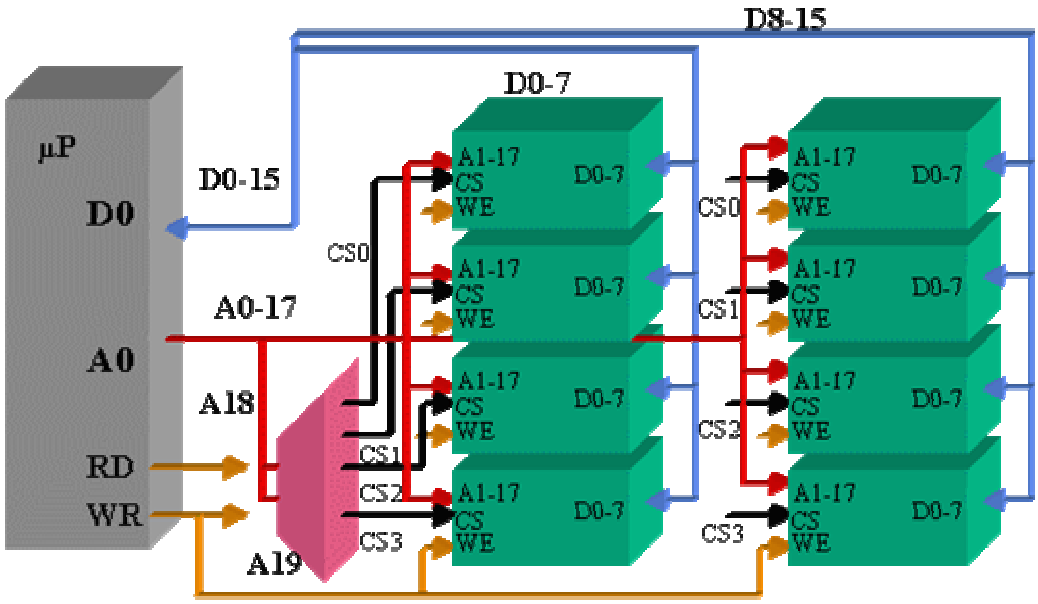


Figure 2.15. Réalisation du plan mémoire.

Le décodeur positionne à 1 (5v) la sortie n° "A1A0". Exemple: A1A0=11b (binaire) = 3d (décimal) =>Y3Y2Y1Y0=(1000)b. Dans le cas de la figure ci dessus les entrées A1 et A0 du décodeur sont les lignes d'adresse A19 et A18 du microprocesseur, ce qui permet de sélectionner 128 Ko par sortie Yi du décodeur (CS1, CS2, CS3 et CS4). La mémoire totale sélectionnée est alors de 512 Ko par banc mémoire. Soit au total 1Mo de cases mémoires sélectionnées.