Chapitre 6 : La machine pédagogique

1/ Généralités et définitions

La machine pédagogique expose les principes de base du traitement programmé de l'information.

La mise en œuvre de cette machine s'appuie sur deux modes de réalisation distincts, le matériel et le logiciel :

- Le matériel (**hardware**) correspond à l'aspect concret du système (unité centrale, mémoire, organes d'entrées-sorties, etc…)
- Le logiciel (**software**) correspond à un ensemble d'instructions, appelé programme, qui sont contenues dans les différentes mémoires du système et qui définissent les actions effectuées par le matériel.

2/ Architecture de base

2.1 Modèle de Von Neumann

John Von Neumann est à l'origine d'un modèle de machine universelle de traitement programmé de l'information **(1946).** Elle est composée des éléments suivants:

- une unité centrale
- une mémoire principale
- des interfaces d'entrées/sorties

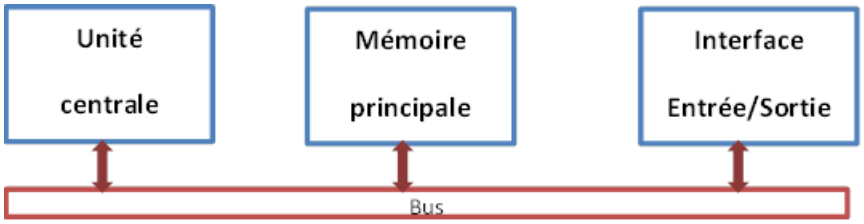


Fig. Modèle de Von Neumann

Les différents organes du système sont reliés par des voies de communication appelées **bus**.

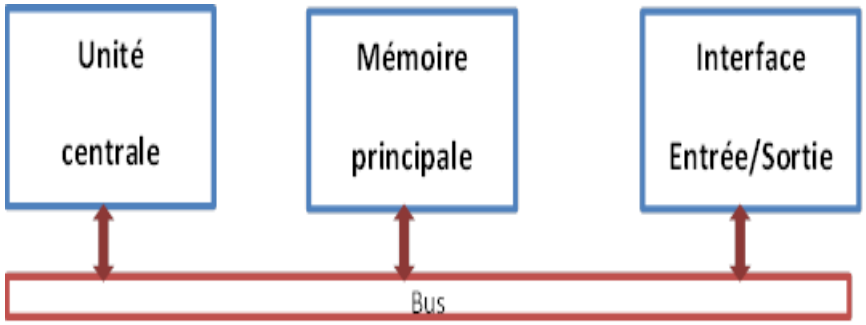


Fig. Modèle de Von Neumann

2.1.1 L'unité centrale

Elle est composée par **le microprocesseur** qui est chargé de :

- ✓ d'interpréter et d'exécuter les instructions d'un programme
- ✓ lire ou de sauvegarder les résultats dans la mémoire et de communiquer avec les unités d'échange

Quelques caractéristiques du microprocesseur:

- ❖ Sa fréquence d'horloge : en MHz ou GHz
- ❖ Le nombre d'instructions par secondes qu'il est capable d'exécuter : en MIPS
- ❖ La taille des données qu'il est capable de traiter : en bits

2.1.2 La mémoire principale :

Elle contient les instructions ou des programmes en cours d'exécution et les données associées à ce programme.

Physiquement, elle se décompose souvent en :

- ✓ une mémoire morte (**ROM** = Read Only Memory) chargée de stocker le programme. C'est une mémoire à lecture seule.

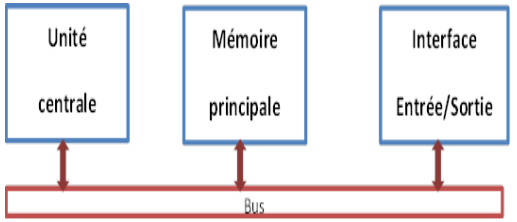
- ✓ une mémoire vive (**RAM** = Random Access Memory) chargée de stocker les données intermédiaires ou les résultats de calculs. On peut lire ou écrire des données dedans, ces données sont perdues à la mise hors tension.

Exemple d'une barrette de RAM réelle

Remarque :
Les disques durs, CDROM, …etc sont des périphériques de stockage et sont considérés comme des mémoires secondaires.

2.1.3 Les interfaces d'entrées/sorties

Elles permettent d'assurer la communication entre le microprocesseur et les périphériques. (capteur, clavier, afficheur, modem, imprimante, etc…).



2.1.4 Les bus

Un bus est un ensemble de fils qui assure la transmission du même type d'information.
On retrouve trois types de bus véhiculant des informations en parallèle dans un système de traitement programmé de l'information:

- **un bus de données** : bidirectionnel qui assure le transfert des informations entre le microprocesseur et son environnement, et inversement. Son nombre de lignes est égal à la capacité de traitement du microprocesseur.

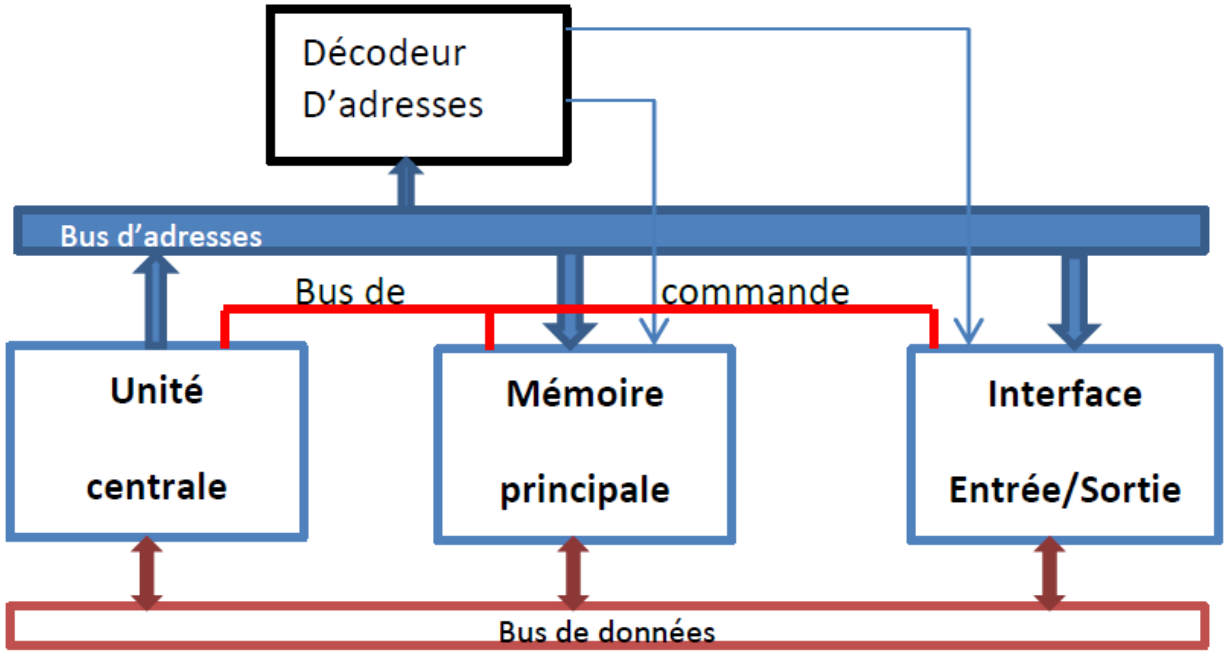
- **un bus d'adresses**: unidirectionnel qui permet la sélection des informations à traiter dans un *espace mémoire* (ou *espace adressable*) qui peut avoir 2^n emplacements, avec n = nombre de conducteurs du bus d'adresses.

- **un bus de commande**: constitué par quelques conducteurs qui assurent la synchronisation des flux d'informations sur les bus des données et des adresses.

2.1.5 Décodage d'adresses

La multiplication des périphériques autour du microprocesseur nécessite la présence d'un **décodeur d'adresse** chargé d'aiguiller les données présentes sur le bus de données.
En effet, le microprocesseur peut communiquer avec les différentes mémoires et les différents boîtiers d'interface qui sont tous reliés sur le même bus de données.

Afin d'éviter des conflits, un seul périphérique doit être sélectionné à la fois. Pour ce faire on attribue à chaque périphérique une zone d'adresse et par conséquent une fonction « décodage d'adresse » est nécessaire afin de fournir les signaux de sélection de chacun des composants.



Remarque :
Lorsqu'un composant n'est pas sélectionné, ses sorties sont mises à l'état « haute Impédance » afin de ne pas perturber les données circulant sur le bus. (elle présente une impédance de sortie très élevée = circuit ouvert).

3/ Le microprocesseur

Un microprocesseur est un circuit intégré complexe caractérisé par une très grande intégration et doté des facultés d'interprétation et d'exécution des instructions d'un programme. C'est le "cerveau" de l'ordinateur

Rôle:

- Il est chargé d'interpréter et d'exécuter les instructions d'un programme
- De lire ou sauvegarder les résultats dans la mémoire
- De communiquer avec les unités d'échange

A l'heure actuelle, un microprocesseur regroupe sur quelques millimètres carrés des fonctionnalités toujours plus complexes. Leur puissance continue de s'accroître et leur encombrement diminue régulièrement .

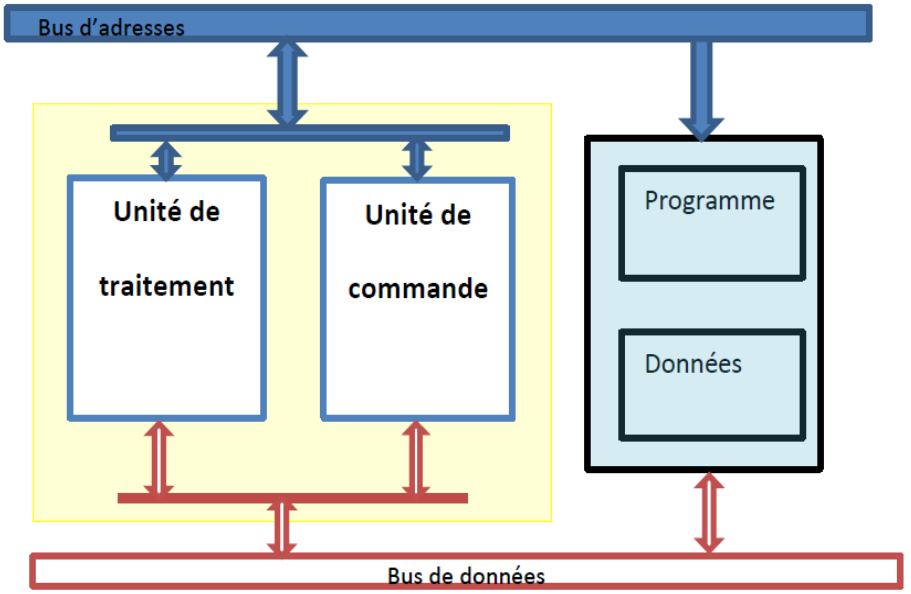
Exemple de microprocesseur

3.1 Architecture de base d'un microprocesseur

Un microprocesseur est construit autour de trois éléments principaux :

- ✓ Une unité de commande
- ✓ Une unité de traitement
- ✓ des registres chargées de stocker les différentes informations à traiter.

Ces trois éléments sont reliés entre eux par des bus internes permettant les échanges d'informations.



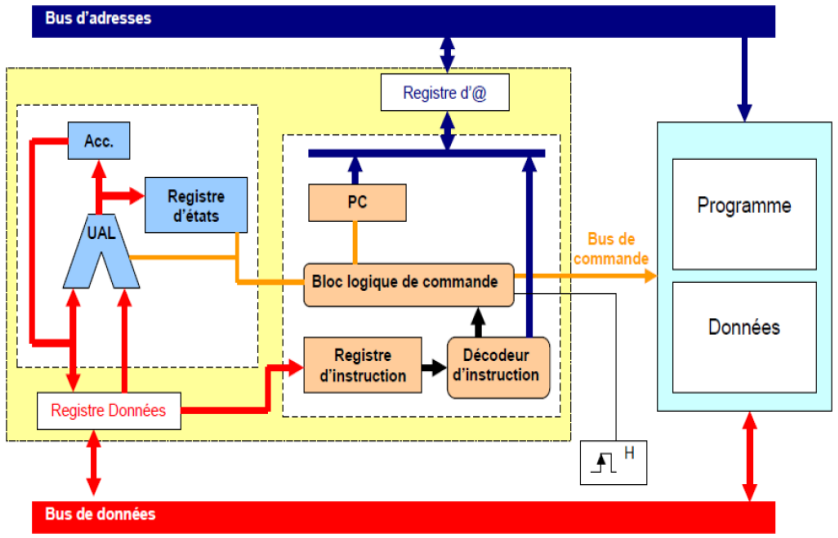
Remarques :
Il existe deux types de registres :

- ✓ les registres d'usage général qui permettent à l'unité de traitement de manipuler des données à vitesse élevée. Ils sont connectés au bus de données interne au microprocesseur.
- ✓ les registres d'adresses (pointeurs) connectés sur le bus adresses.

3.1.1 L'unité de commande

Elle permet de séquencer le déroulement des instructions.
elle est composée par :

Schéma fonctionnel d'un microprocesseur



- ❖ **le compteur de programme (PC)** appelé aussi **le compteur ordinal (CO)** constitué par un registre dont le contenu est initialisé avec l'adresse de la première instruction du programme. Il contient toujours l'adresse de l'instruction à exécuter.

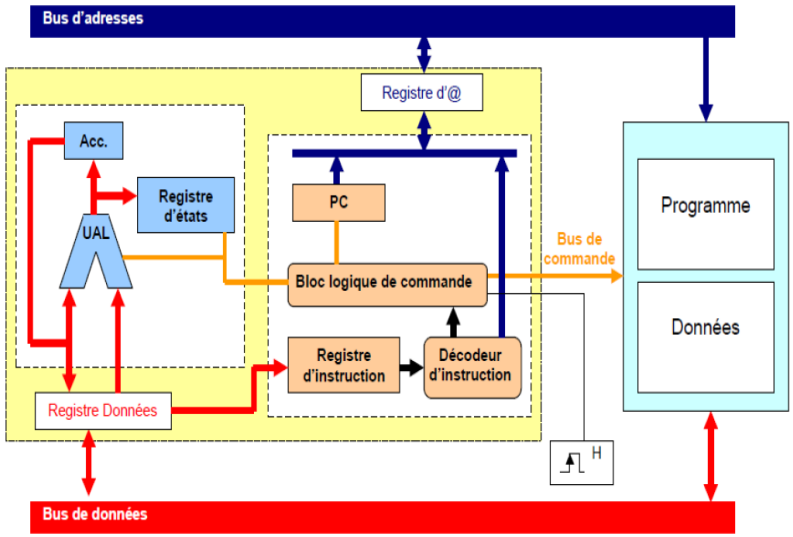
- ❖ **le registre d'instruction et le décodeur d'instruction :** chacune des instructions à exécuter est rangée dans le registre d'instruction puis est décodée par le décodeur d'instruction.

- ❖ **Bloc logique de commande (ou séquenceur) :** Il organise l'exécution des instructions au rythme d'une horloge. Il élabore tous les signaux de synchronisation internes ou externes **(bus de commande)** du microprocesseur en fonction des divers signaux de commande provenant du décodeur d'instruction ou du registre d'état par exemple. Il s'agit d'un automate réalisé de façon micro-programmée.

3.1.2 L'unité de traitement

C'est le cœur du microprocesseur. Elle regroupe les circuits qui assurent les traitements nécessaires à l'exécution des instructions :

Schéma fonctionnel d'un microprocesseur



✓ L'Unité Arithmétique et Logique (UAL)

C'est un circuit complexe qui assure les fonctions logiques (ET, OU, Comparaison, Décalage , etc…) ou arithmétique (Addition, soustraction…etc).

✓ Le registre d'état

- Il est généralement composé de 8 bits à considérer individuellement. Ils s'appellent indicateurs (drapeaux ou flags).
- Ces indicateurs sont mis a jours après la fin de l'exécution d'une opération dans l'UAL.
- les principaux indicateurs sont :

❖ **retenue** : ce bit est mis à 1 si l'opération génère une retenue
❖ **signe** : ce bit est mis à 1 si l'opération génère un résultat négatif.
❖ **débordement** : ce bit est mis à 1 s'il y a un débordement
❖ **zéro** : Ce bit est mis à 1 si le résultat de l'opération est nul.

✓ Les accumulateurs

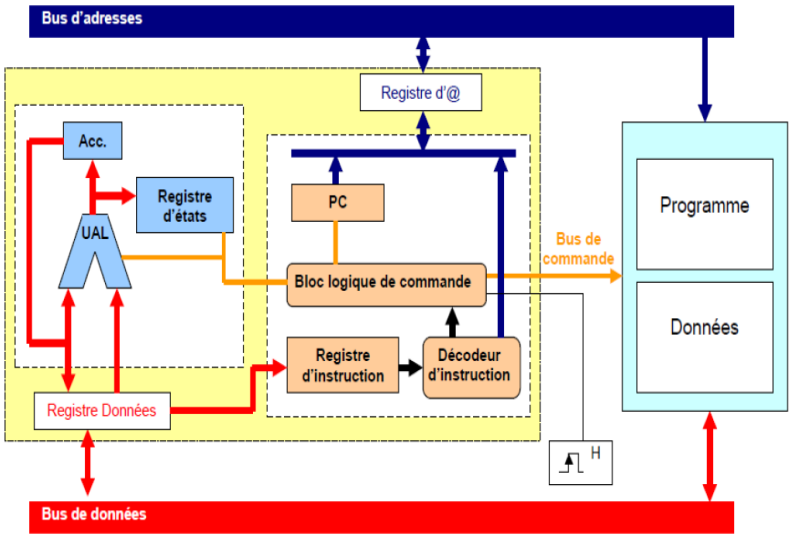
ce sont des registres de travail qui servent à stocker une opérande au début d'une opération arithmétique et le résultat à la fin de l'opération.

3.2 Cycle d'exécution d'une instruction

Les instructions sont codées en binaire. Leur traitement, par le microprocesseur, peut être décomposé en deux cycles.

➢ **Cycle 1:** *Recherche de l'instruction à traiter* **(voir schéma ci-contre)**

Schéma fonctionnel d'un microprocesseur



1) Transfert de l'adresse de l'instruction du CØ (PC) vers le MAR (Registre d'@).
2) L'impulsion de lecture générée par l'unité de commande provoque le transfert de l'instruction vers le MBR(registre données).
3) Transfert de l'instruction du MBR vers le Registre d'Instruction(RI).

L'instruction a la forme suivante :

Code opération + adresse opérande

4) L'adresse opérande est envoyée vers le MAR

➢ Le code opération est transmis au décodeur d'instruction pour déterminer l'opération à effectuer.

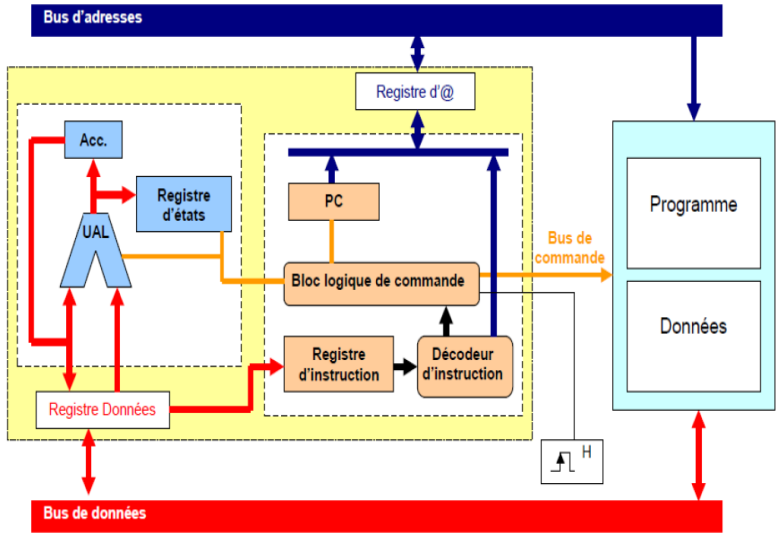
➢ L'opération décodée est transmise au séquenceur pour être réalisée.

5) Le CØ (PC) est incrémenté pour le cycle recherche suivant.

➢ **Cycle 2 : Exécution (voir schéma ci-contre)**

Le cycle recherche est suivi par le cycle exécution durant lequel l'opération spécifiée par l'instruction est exécutée par l'UAL.
Les étapes sont les suivantes : **(voir schéma ci-contre)**

Schéma fonctionnel d'un microprocesseur



1) Le séquenceur envoie des signaux de commande vers la mémoire centrale (MC) pour lire l'opérande dont l'adresse est déjà dans le MAR.

2) L'opérande est transféré de la RAM vers le MBR(Registre données)

3) Le contenu du MBR est transféré vers l'accumulateur.

4) L'opération est effectuée par un ensemble de circuits dans l'UAL sous le contrôle du séquenceur.

A la fin du cycle d'exécution, l'unité de commande passe immédiatement au cycle recherche de l'instruction suivante.

3.3 Jeu d'instructions

3.3.1 Définition

La première étape de la conception d'un microprocesseur est la détermination de son jeu d'instructions. Le jeu d'instructions décrit l'ensemble des opérations élémentaires que le microprocesseur pourra exécuter. Il va donc en partie déterminer l'architecture du microprocesseur à réaliser et notamment celle du séquenceur.

Remarque:
A un même jeu d'instructions peut correspondre un grand nombre d'implémentations différentes du microprocesseur.

3.3.2 Type d'instructions

Les instructions que l'on retrouve dans chaque microprocesseur peuvent être classées en 4 groupes :

- ✓ **Transfert de données** pour charger ou sauver en mémoire, effectuer des transferts de registre à registre, etc…
- ✓ **Opérations arithmétiques** : addition, soustraction, division, multiplication
- ✓ **Opérations logiques :** ET, OU, NON, NAND, comparaison, test, etc…
- ✓ **Contrôle de séquence :** branchement, test, etc…

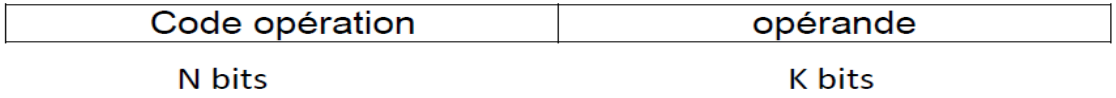
3.3.3 Codage d'une instruction

Les instructions et leurs opérandes (données) sont stockés dans la mémoire.

La taille d'une instruction (nombre de bits nécessaires pour la représenter en mémoire) dépend du type de l'instruction et du type de l'opérande.

L'instruction est découpée en deux parties :

- ✓ **Code opération (code instruction) : un code sur N bits qui indique quelle instruction.**
- ✓ **Le champ opérande : qui contient la donnée ou la référence (adresse) à la donnée.**



Remarque:
Le format d'une instruction peut ne pas être le même pour toutes les instructions. Le champ opérande peut être découpé à son tour en plusieurs champs

Machine à 2 adresses

Dans ce type de machine pour chaque instruction il faut préciser :

- l'adresse du premier opérande
- Celle du deuxième opérande,

Code opération	Opérande1	Opérande2

l'adresse du résultat est implicitement l'adresse du deuxième opérande.

Exemple :

ADD A, B (B←A+B)

Machine à 1 adresse

Dans ce type de machine pour chaque instruction il faut préciser uniquement l'adresse du deuxième opérande.
- Le premier opérande existe dans le registre accumulateur.
- Le résultat est mis dans le registre accumulateur

Code opération	Opérande 2

Exemple :

ADD A (ACC← (ACC)+A)

Ce type de machine est le plus utilisé.

Mode d'adressage:

Le mode d'adressage définit la manière dont le microprocesseur va accéder à l'opérande.
Le champ opérande contient la donnée ou la référence (adresse) à la donnée.
Le code opération de l'instruction comporte un ensemble de bits pour indiquer le mode d'adressage.
Les modes d'adressage les plus utilisés sont :

- ✓ Immédiat
- ✓ Direct
- ✓ Indirect
- ✓ Indexé
- ✓ Relatif

Adressage immédiat :

L'opérande existe dans le champ adresse de l'instruction

Code opération	Opérande

Exemple :

ADD 150

ADD	150

Cette commande va avoir l'effet suivant : ACC← (ACC)+150

Si le registre accumulateur contient la valeur 200 alors après l'exécution, son contenu sera égal à 350.

Adressage direct :

Le champ opérande de l'instruction contient l'adresse de l'opérande (emplacement en mémoire). Pour réaliser l'opération il faut récupérer (lire) l'opérande à partir de la mémoire.

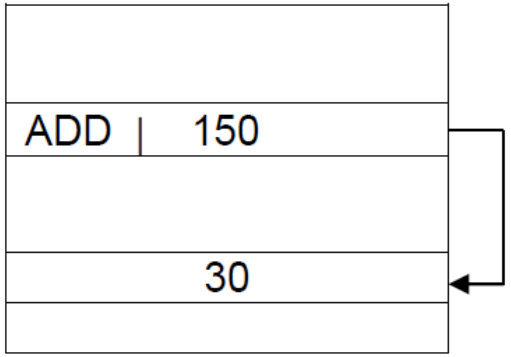
Exemple :

ADD 150

ACC ←(ACC) + (ADR)

Exemple :

On suppose que l'accumulateur contient la valeur 20.
A la fin de l'exécution nous allons avoir la valeur
50 (20 +30)



Adressage indirect

Le champ adresse contient l'adresse de l'adresse de l'opérande. Pour réaliser l'opération il faut :

- **Récupérer l'adresse de l'opérande à partir de la mémoire.**
- **Par la suite il faut chercher l'opérande à partir de la mémoire.**

AC← (ACC) + ((ADR))

Exemple :

Initialement l'accumulateur contient la valeur 20

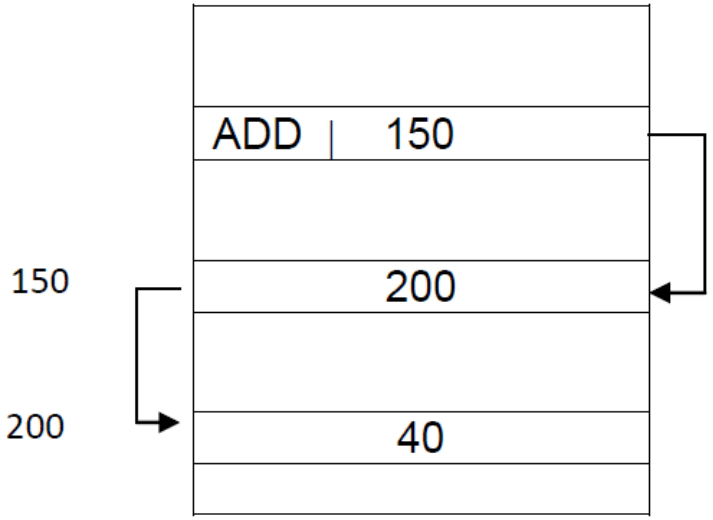
• Il faut récupérer l'adresse de l'adresse (150).

• Récupérer l'adresse de l'opérande à partir de

l'adresse **150** (**la valeur 200**)

• Récupérer la valeur de l'opérande à partir de l'adresse

200 (**la valeur 40)**



• Additionner la valeur 40 avec le contenu de l'accumulateur (20) et nous allons avoir la valeur **60**

Adressage indexé

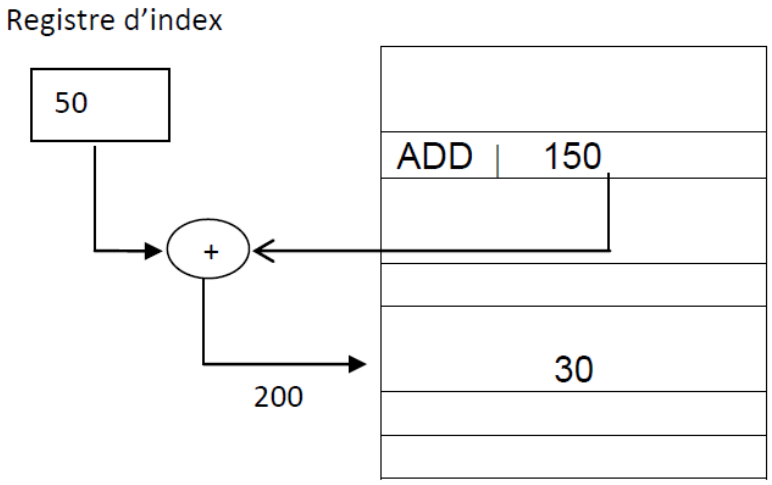
• L'adresse effective de l'opérande est relative

à une zone mémoire.

• L'dresse de cette zone se trouve dans un registre spécial

(registre indexe).

• Adresse opérande = ADR + (X)



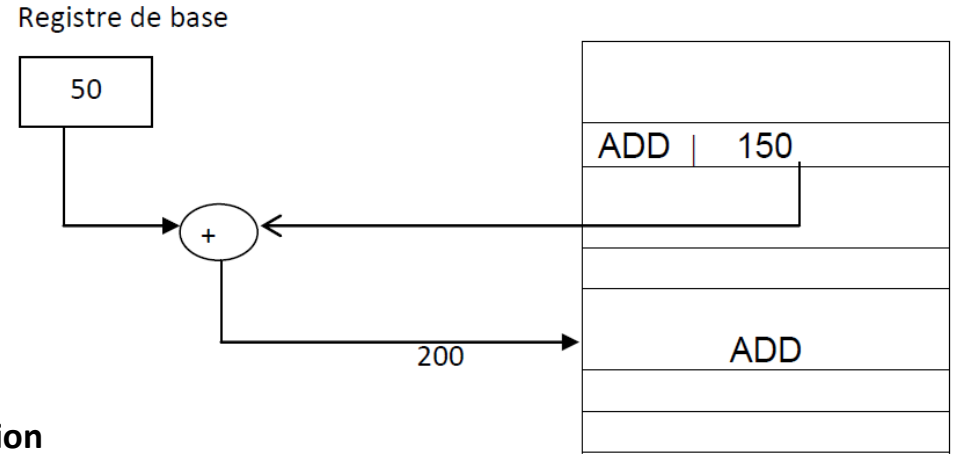
Exemple : ADD 150, X

On suppose que l'accumulateur contient la valeur 20.
A la fin de l'exécution nous allons avoir la valeur
50 (20 +30)

Adressage relatif

- L'adresse effective de l'opérande est relative à une zone mémoire.
- L'dresse de cette zone se trouve dans un registre spécial (registre de base exp: PC).
- Ce mode d'adressage est utilisé pour les instructions de branchement.

Adresse = ADR + (base)



Cycle d'exécution d'une instruction

Le traitement d'une instruction est décomposé en trois phases :

- Phase 1 : recherche de l'instruction à traiter et décodage
- Phase 2 : recherche de l'opérande et exécution de l'instruction
- Phase 3 : passage à l'instruction suivante

• Chaque phase comporte un certain nombre d'opérations élémentaires (microcommandes) exécutées dans un ordre bien précis (elles sont générées par le séquenceur).

• Les phases 1 et 3 ne changent pas pour l'ensemble des instructions, par contre la phase 2 change selon l'instruction et le mode d'adressage.

Exemple1 *: déroulement de l'instruction d'addition en mode immédiat*

ACC ←(ACC) + Valeur

– Phase 1 : (rechercher l'instruction à traiter)

- Mettre le contenu du CO dans le registre MAR : MAR ←(CO)
- Commande de lecture à partir de la mémoire
- Transfert du contenu du MBR dans le registre RI : RI ←(MBR)
- Analyse et décodage

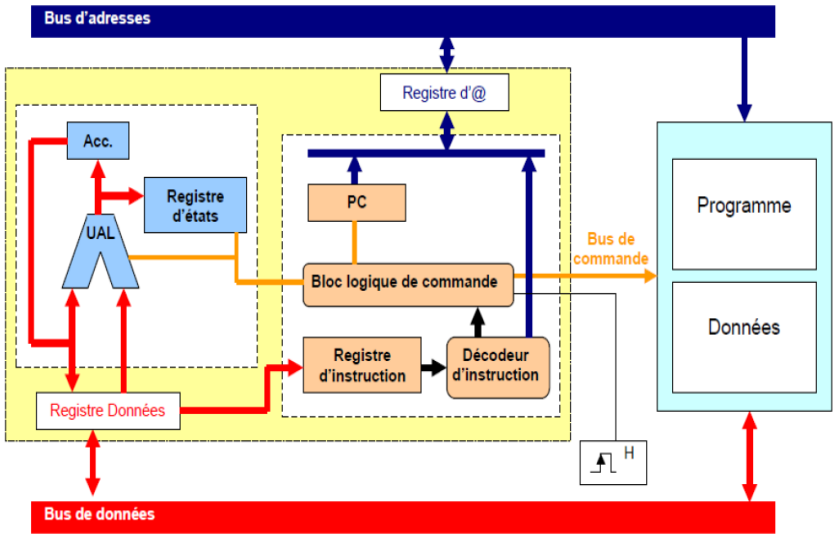
- Phase 2 : (traitement)

- Transfert de l'opérande dans l'UAL : UAL←(RI)
- Commande de l'exécution de l'opération (addition)

– Phase 3 : (passer à l'instruction suivante)

- CO ←(CO) + 1

Schéma fonctionnel d'un microprocesseur



Exemple 2 *: déroulement de l'instruction d'addition en mode direct*

AC ←(ACC) + (ADR)

– **Phase 1 : (rechercher l'instruction à traiter)**

- Mettre le contenu du CO dans le registre MAR : MAR ←(CO)
- Commande de lecture à partir de la mémoire
- Transfert du contenu du MBR dans le registre RI : RI←(MBR)
- Analyse et décodage

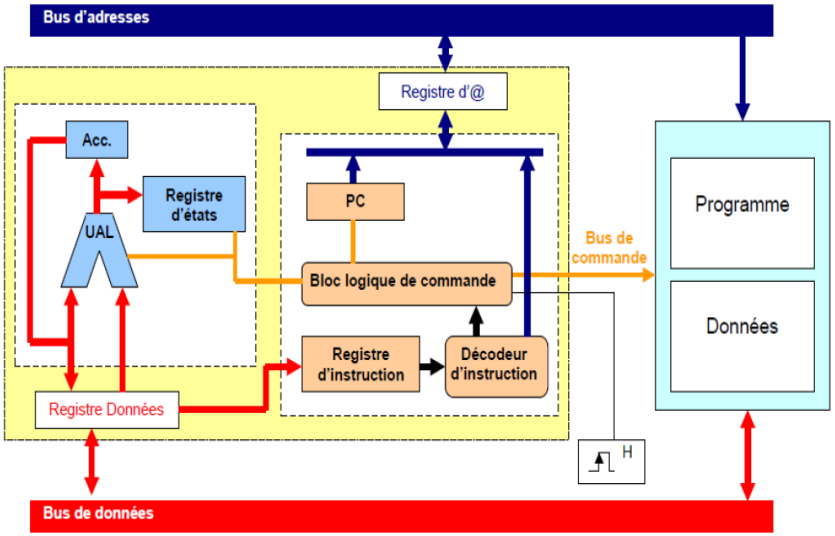
– **Phase 2 : (décodage et traitement)**

- Transfert de l'adresse de l'opérande dans le MAR : MAR ←(RI)
- Commande de lecture
- Transfert du contenu du MBR vers l'UAL : UAL ←(MBR)
- Commande de l'exécution de l'opération (addition)

– **Phase 3 : (passer à l'instruction suivante)**

- CO ←(CO) + 1

Schéma fonctionnel d'un microprocesseur



Exemple 3 *: Déroulement de l'instruction d'addition en mode indirect*

ACC ←(ACC) + ((ADR))

– Phase 1 : (rechercher l'instruction à traiter)

- Mettre le contenu du CO dans le registre MAR : MAR ←(CO)
- Commande de lecture à partir de la mémoire
- Transfert du contenu du MBR dans le registre RI : RI←(MBR)
- Analyse et décodage

– Phase 2 : (décodage et traitement)

Transfert de l'adresse de l'opérande dans le MAR : MAR←(RI).ADR

- Commande de lecture /* récupérer l'adresse */
- Transfert du contenu du MBR vers le MAR : MAR ←(MBR)
- Commande de lecture /* récupérer l'opérande */
- Transfert du contenu du MBR vers l'UAL : UAL←(MBR)
- Commande de l'exécution de l'opération (addition)

– Phase 3 : (passer à l'instruction suivante)

CO ←(CO) + 1

Schéma fonctionnel d'un microprocesseur