CHAPITRE III : Introduction au microprocesseur

III.1 Architecture interne d'un microprocesseur

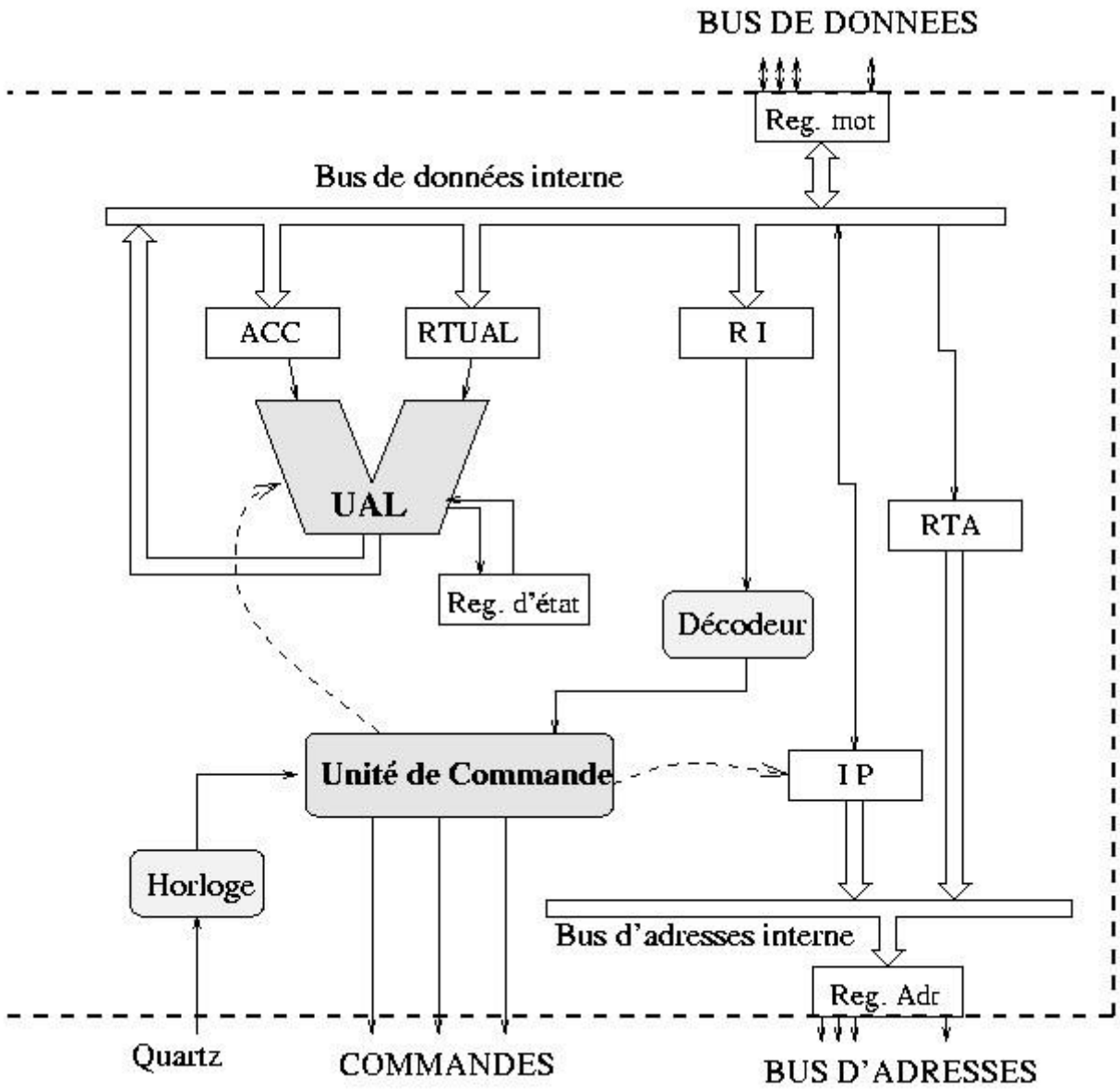


Figure VI.1 Schéma simplifié d'un processeur.

La figure représente l'architecture interne simplifiée d'un microprocesseur, elle comporte:

- Les regitres
- L'*unité de commande*,
- L'*UAL*, les registres
- Le *décodeur* d'instructions, qui, à partir du code de l'instruction lu en mémoire actionne la partie de l'unité de commande nécessaire.

Les informations circulent à l'intérieur du processeur sur deux *bus internes*, l'un pour les données, l'autre pour les instructions.

Le p est relié à l'extérieur par les bus de données et d'adresses, le signal d'horloge et les signaux de commandes.

III.1.1 Les registres

Le processeur utilise toujours des *registres*.

Ce sont des petites mémoires internes d'accès très rapides.
Elles sont utilisées pour stocker temporairement une donnée, une instruction ou une adresse.

Chaque registre stocke 8, 16 ou 32 bits.

Parmi les registres, le plus important est le registre *accumulateur*.

L'accumulateur est utilisé pour stocker les résultats des opérations arithmétiques et logiques.

Un accumulateur est un registre lié d'un coté à un registre temporaire et au bus interne de données et de l'autre coté à la sortie de l'UAL.

Déroulement d'une opération traitant deux données :

1. Une donnée sera stockée tout d'abord dans l'accumulateur
2. Cette donnée passera en suite au premier registre temporaire
3. La deuxième donnée passera directement au deuxième registre temporaire.

4.Le résultat sera stocké dans l'accumulateur par l'intermédiaire de la sortie de l'UAL.

Exemple : addition de deux données : D1 et D2

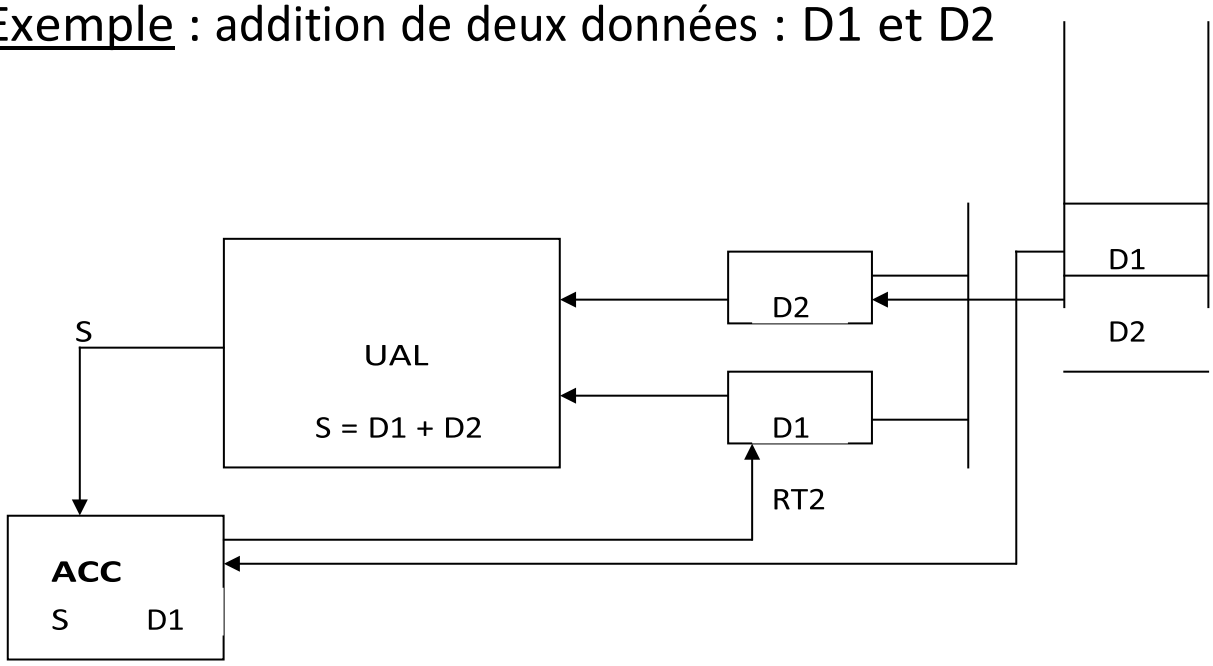
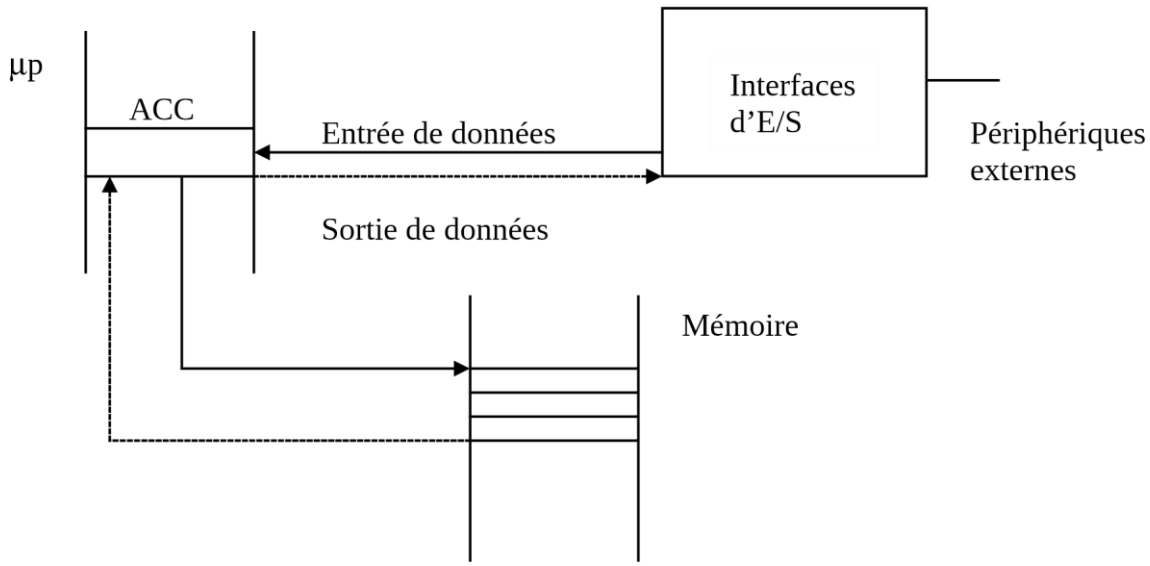


Figure III.4 Addition de deux données

Un accumulateur représente également une intermédiaire entre la mémoire et les interfaces d'E/S



Le nombre de bits de l'ACC est égal en général au nombre de broches du bus de données.

Le nombre d'accumulateur varie d'un µp à l'autre.

Exemple : première génération : 8088, 8085 et 6800 ce sont des µp de 8 bit ils possèdent 1 accumulateur de 8 bits.

6809 et 8088 sont appelés des µps faux 16 bits :

- ce sont des µps de 8 bits ;
- Ils possèdent deux accumulateur de 8 bits qui peuvent être regroupés pour former un seul accumulateur de 16 bits.

Le 8086, 80286, 68000 sont de vrais 16 bits : Ce sont des µps de 16 bits avec 1 accumulateur de 16 bits.

Les µps vrais 16 bits sont plus rapides que les µps faux 16 bits.

L'accumulateur intervient dans une proportion importante des instructions.

Exemple : Exécution d’une instruction comme ``*Ajouter 5 au contenu de la case mémoire d'adresse 180*'' :

1. Le processeur lit et décode l'instruction;
2. le processeur demande à la mémoire le contenu de l'emplacement 180;
3. la valeur lue est rangée dans l'accumulateur;
4. l'unité de traitement (UAL) ajoute 5 au contenu de l'accumulateur;
5. le contenu de l'accumulateur est écris en mémoire à l'adresse 180.

C'est l'unité de commande qui déclenche chacune de ces actions dans l'ordre.

L'addition proprement dite est effectuée par l'UAL.

Il existe aussi six registres fondamentaux qu’on trouve dans tous les p :

- Compteur d’instructions ou compteur ordinal
- Registre d’adresse
- Registre d’instruction
- Registre de données (accumulateur)
- Registre d’Etat

- Registre temporaire

Selon le type du p utilisé on peut trouver en plus de ces registres d'autres registres pour faciliter la programmation d'un p.

III.1.1.1 Compteur ordinal (CO)

Format d'une instruction

Chaque instruction est représentée généralement par deux champ :

code opération : spécifie au p la nature de l'opération effectuée.

Cette opération peut être une +, -, *, un transfert de données, ….

Le code opération est codé sur une cellule mémoire.

Code opérande : indique au p les données qui vont être traitées par le code opération.

La taille du code opérande dépend de la nature des données mises en jeu.

Un certain nombre de cellules mémoire est nécessaire pour stocker le code opérande.

La structure d'un programme stocké en mémoire a la forme suivante :

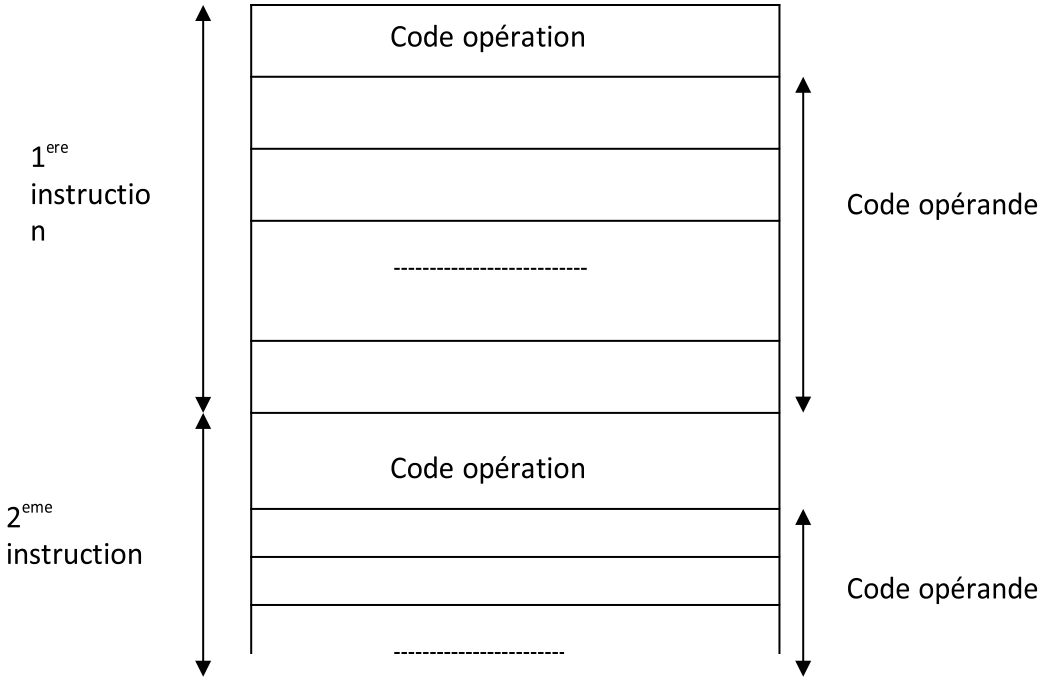


Figure III.2 structure d'un programme en mémoire

Le compteur ordinal est un registre qui permet de localiser les instructions stockées en mémoire.

Le CO est chargé au début de l'exécution d'un programme à partir de l'adresse de la première case

mémoire où se trouve le premier code opération de la première instruction.

L'exécution d'un programme s'effectue d'une façon séquentielle.

le compteur ordinal s'incrémente automatiquement pour passer de l'adresse d'une case mémoire à l'adresse suivante.

Le nombre de bits de ce registre est égal au nombre de broches du registre d'adresse.

III.1.1.2 registre d'adresse (RA)

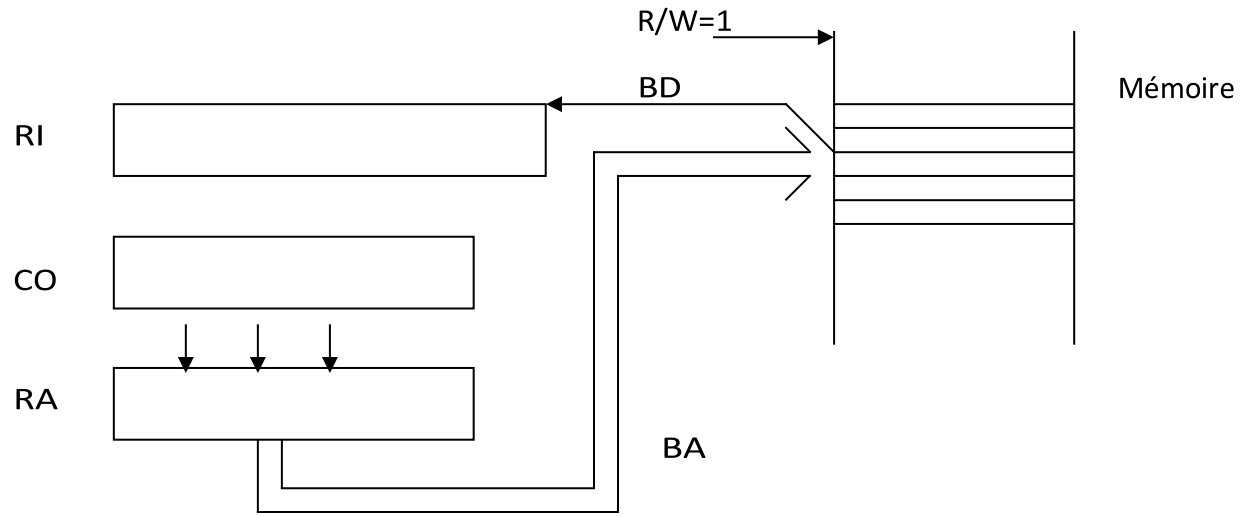
Le RA permet d'accéder à une donnée en mémoire.

Le RA a le même nombre de bits que le compteur ordinal.

III.1.1.3 registre d'instruction (RI)

Le RI permet de stocker l'instruction à exécuter.

Le RI garde l'instruction pendant le temps nécessaire à son décodage et à son exécution.



Le compteur ordinal communique avec le RA pour lui donner l'adresse de la première instruction.

Le RA donne cette adresse au bus d'adresse qui va localiser cette adresse.

La mémoire reçoit un signal (la ligne R/W =1) de la part de la logique de contrôle pour l'informer qu'il s'agit d'une lecture.

Le bus de données va lire l'instruction et la mettre dans le registre d'instruction.

III.1.1.4 registres temporaires

L'UAL possède deux entrées pour recevoir les données à traiter.

Au niveau de chaque entrée il y a un registre temporaire qui garde la donnée pendant le temps nécessaire à son traitement.

Exemple : traitement de deux données

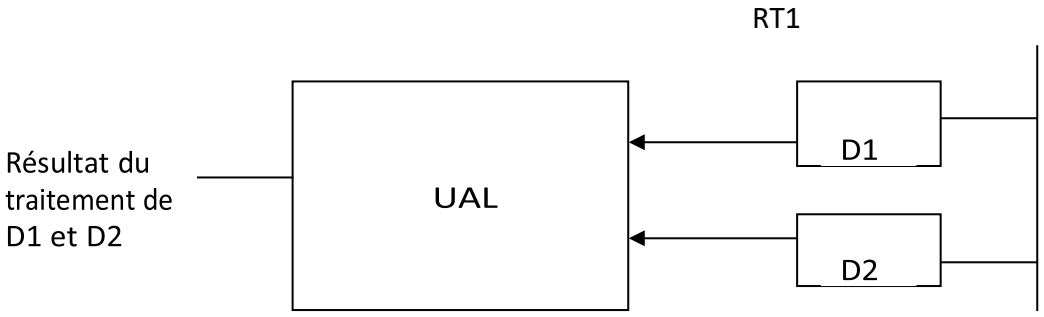


Figure III.3 traitement de deux données par l'UAL$_{RT2}$

L'UAL est un circuit électronique qui ne peut pas stocker des données, elle se sert donc des registres temporaires RT1 et RT2.

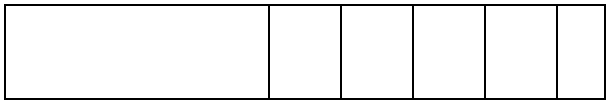
III.1.1.5 Registre d'Etat

Le registre d'Etat est directement lié à l'UAL.

Le registre d'Etat stocke certaines informations particulières concernant les opérations effectuées.

Le nombre de bits de ce registre varie d'un ?p à l'autre.

Les bits les plus utilisés sont :



Bit de retenu ou indicateur de retenu (CF)

Si CF = 1 alors une opération d'addition ou soustraction a généré une retenue,

Si CF = 0 alors pas de retenue

Exemples

a.

10101111
00001010
10101001

CF=0 pas de retenue

b.

11111111 (255)
00000001 (1)
00000000

CF=1 il y a une retenue, ce 1 de retenue sera stocker dans le bit de retenue CF.

Bit de zéro ZF

Si ZF = 1alors le résultat d'une opération est nul.

Si ZF = 0 alors le résultat d'une opération est non nul.

Bit de signe (SF)

Si SF = 1 alors le bit le plus significatif d'une donnée est égal à 1

Si SF = 0 alors le bit le plus significatif d'une donnée est égal à 0

Exemple : en complément à 1 ou à 2 la donnée 10000001 implique SF = 1.

Bit de débordement OF

Si OF = 1 alors il y a dépassement de la capacité maximale de codage.

Si OF = 0 alors il n'y a dépassement de la capacité maximale de codage.

III.1.2 L'unité de traitement (UAL)

Le traitement effectué par l'unité arithmétique et logique sont +, -, *, /, le et logique, le ou logique, le ou exclusif. Exemples :

a. le et logique (AND)

Soient N1 = 00101111 et N2 = 10100001 N3 = N1 AND N2 = 00100001

Table logique de AND :

A	B	A AND B
0	0	0
1	0	0
0	1	0
1	1	1

b. le ou logique (OR)

Table logique de OR :

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

c. le ou exclusif (EOR) table

logique de EOR

A	B	A EOR B
0	0	0

0	1	1
1	0	1
1	1	0

III.1.3 Unité de commande et de contrôle

L'unité de commande est le maître d'œuvre des différentes opérations de transfert de données.

L'unité de commande a pour fonction :

- Décodage des instructions stockées dans le registre d'instructions (nature de l'opération ou l'instruction à exécuter).

- Exécution des instructions par les organes exécutifs (UAL, Accumulateur, CO, …)
- Gestion du dialogue entre le p, les mémoires et les interfaces d'E /S, par l'intermédiaire du bus de commande et des signaux de contrôle.

Les signaux de commandes permettent au processeur de communiquer avec les autres circuits de l'ordinateur.

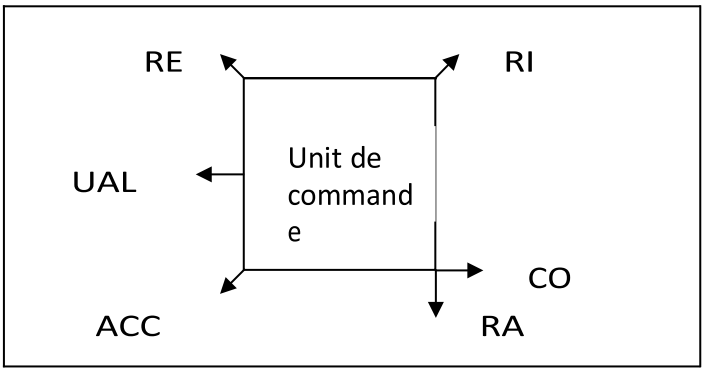
On trouve en particulier :

1.le signal R/W (Read/Write) : le p indique à la mémoire principale s'il effectue un accès en lecture ou en écriture.

2.l'initialisation des registres internes du p : la broche d'initialisation fait signal au p de faire tous les registres à zéro.

3. l'interruption d'exécution du p.

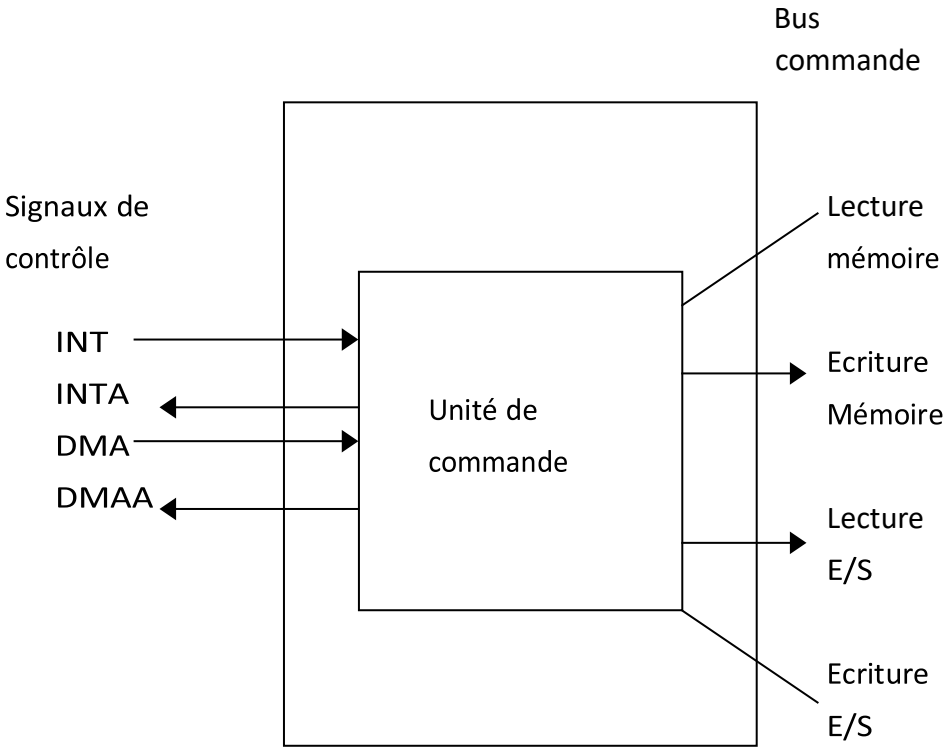
Le nombre de broches de ces signaux change d'un constructeur à un autre.



III.1.3.1 Exemple de gestion interne

- Incrémentation du CO
- Mise à jour des bits du registre d'Etat en fonction des résultats fournis par l'UAL
- Nature de l'opération qui va être effectuée parl'UAL

III.1.3.2 Exemple de gestion externe (dialogue)



- INT : interruption d'exécution d'un programme.

  INT est générée par une interface d'E/S

  Si INT = 1 alors l'IE/S désire communiquer avec le p.

  Si INT = 0 pas de dialogue entre E/S et p

- INTA : Réponse par le p au signale INT

  Si INTA = 1 alors le p a reçu le signal INT

Si INTA = 0 alors le □p n'a pas reçu le signal INT

- DMA : Direct Memory Acces

  Si DMA = 1 alors une interface d'E/S demande les bus de données, d'adresses et de commandes pour être en lien avec la mémoire.

  Si DMA = 1 alors pas de demande des bus de la part les interfaces d'E/S.

- DMAA : réponse du □p au signal DMA

  Si DMAA = 1 alors le □p a reçu le signal DMA

  Si DMAA = 0 alors le □p n'a pas reçu le signal DMA

<u>Exemple d'utilisation</u>

- Chargement des programmes, fichiers de données, …, de l'unité disque dur ou disquette dans la mémoire vive (RAM).

- Stockage des programme, … de la RAM vers le disque dur ou disquette.

Exemple de INT et INTA

Si INT = 0, INTA = 0

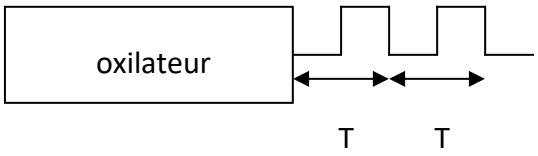
Le p exécute le programme principal (fonctionnement normal)

Si INT = 1 alors une interface d'E/S désire communiquer avec le p pour un transfert de données :

1. Le p va donc interrompre le programme principal
2. INTA = 1, le p informe l'interface d'E/S qu'il a reçu le signal INT
3. Le p exécute un sous programme pour le traitement de l'interruption (lecture des données de l'interface d'E/S vers le p et du p vers la mémoire).
4. A la fin de l'exécution du sous programme, le p continue à exécuter le programme principal à partir de l'instruction suivante.

III.1.4 Horloge

C’est un oscillateur qui génère un signal périodique.



à l’instant T le p doit chercher l’instruction après une période il doit l’exécuter et ainsi de suite.

d’où la notion de temps pour p.

III.2 Fonctionnement d’un p

L’exécution d’un programme se fait séquentiellement à partir du premier mot de la première instruction.

Pour chaque mot d’une instruction le p effectue les phases suivantes :

- Phase de recherche
- Phase de décodage et d’exécution

- ## III.2.1 phase de recherche

le but de cette phase est de chercher le contenu d'un mot d'une instruction de la mémoire vers le registre d'instruction.

Elle s'effectue de la manière suivante :

1. Le contenu du compteur ordinal, qui contient l'adresse du premier mot de l'instruction est envoyé au registre d'adresse.

2. Un signal de lecture est envoyé par le bloc de commande et de contrôle vers la mémoire.

3. La mémoire une fois qu'elle reçoit l'adresse et le signal de lecture :

- effectue le décodage de l'adresse considérée (càd cherche la case mémoire dont l'adresse est dans le bus d'adresse).
- dépose sur le bus de données le contenu de la case mémoire spécifiée.

4. Le p lit le bus de données et dépose le contenu dans le registre d'instruction.

5.Le contenu du compteur ordinal est incrémenté automatiquement par le bloc de contrôle et de commande.

Les étapes de 1 à 5 vont se répéter jusqu'à la fin de la recherche de tous les mots de l'instruction.

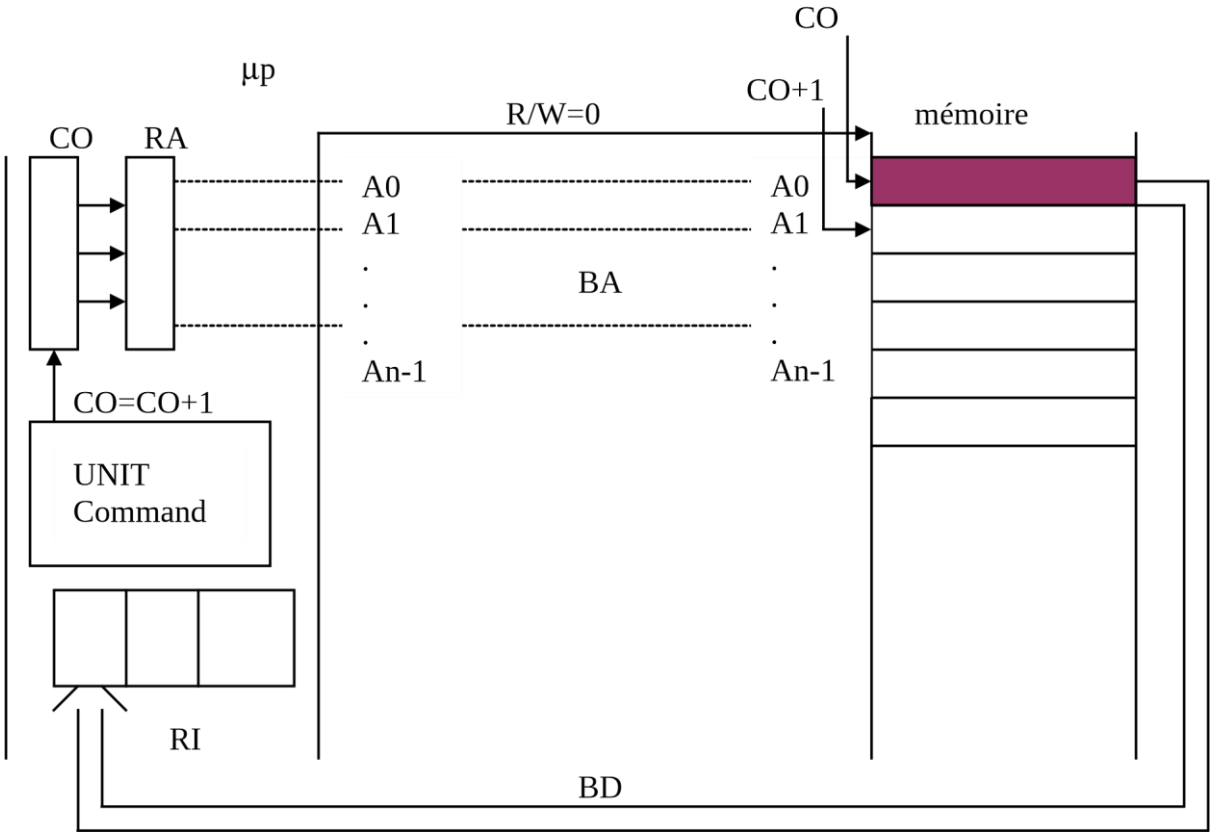


Figure IV.5 phase de recherche d'un cycle machine

III.2.2 Phase de codage

L'instruction est stockée dans le registre d'instruction.

Le bloc de commande et de contrôle procède à son décodage et son exécution.
Le processus (recherche, décodage, et exécution) se répète avec l'instruction suivante dont l'adresse de son premier mot est déjà stockée dans le CO.

La synchronisation entre la phase de recherche, la phase de décodage et d'exécution se fait grâce à une base de temps fournis par un oscillateur.

Un oscillateur est un circuit électronique oscillant.

Le nombre de périodes par phase dépend du p utilisé.

Le nombre de phases de recherche dépend de l'instruction à exécuter.

Le nombre de phases de décodage est 1.

Exemple

LOAD ACC $2000

Càd charger dans l'accumulateur le contenu de la case mémoire : $2000.

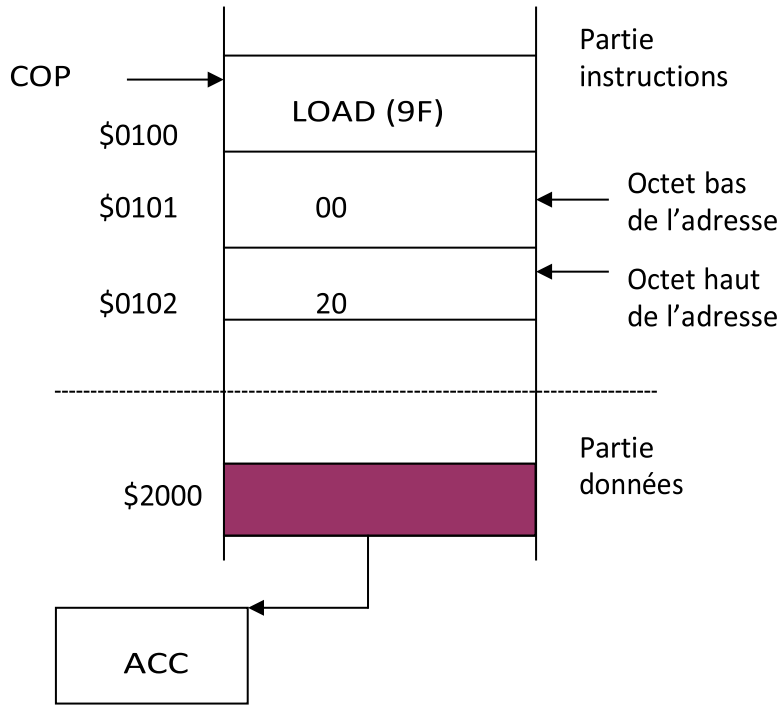


Figure IV.6 phase de codage d'un cycle machine

Phase de recherche : 3 périodes

Phase d'exécution : 2 périodes

Un cycle machine : 5 période

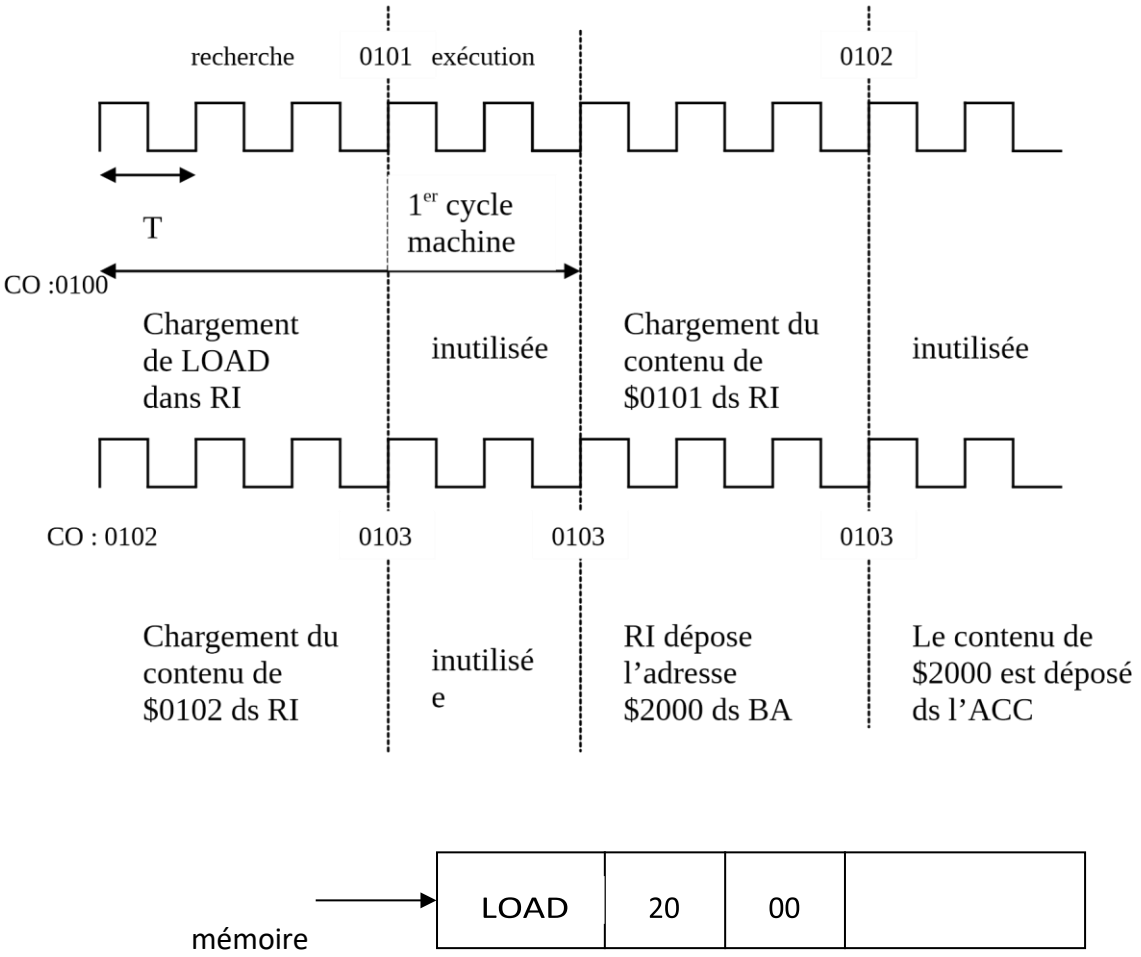


Figure III.7 phases de recherche et d'exécution de l'instruction LOAD ACC 2000

L'adresse du CO s'incrémente à la fin de chaque phase de recherche d'adresse d'instruction et pas d'adresse de données.