Cours Architecture des ordinateurs
Microprocesseurs

Resp. MCA Mohamed Feredj
Courriel : archiFeredj@gmail.com

Plan :

I) Généralité
- a. Bref historique des ordinateurs
- b. Représentation de l'information
- c. Codage des données
  - i. Nombres
  - ii. Caractères
- d. Opérations arithmétiques
- e. Les couches du système informatique

II) **Microprocesseurs**
- a. Structure Abstraite des calculateurs
- b. Circulation de l'information dans un calculateur
- c. Structure de la mémoire centrale
- d. Description matérielle d'un µP (cas du 8086)
  - i. Vue externe
  - ii. Vue interne
- e. Fonctionnement des µP
- f. Amélioration des performances des µP

III) Langages Assembleur et machine

IV) Les interruptions

V) **Les E/S**
- Techniques d'interfaçage
- Par polling /* par programmation */
- Par chaînage
- Par its

VI) Circuits d'interface
- a. Contrôleur d'its /* PIC 8259 */
- b. Timer

VII) La technologie USB

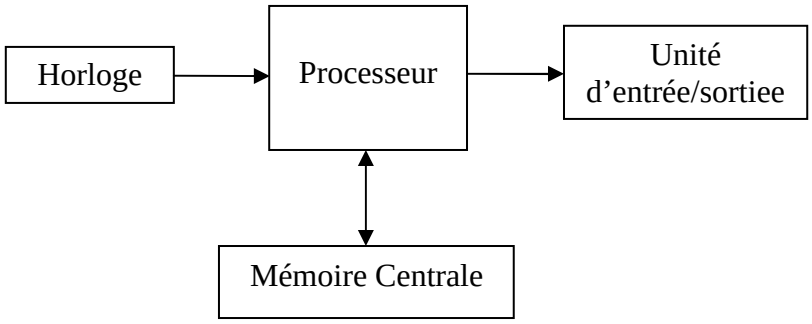
Cours Architecture des ordinateurs
Microprocesseurs

Resp. MCA Mohamed Feredj
Courriel : archiFeredj@gmail.com

1) Structure abstraite des ordinateurs

Def. d'un Ordinateur :

- Est une machine capable de recevoir, d'emmagasiner, de traiter et de transmettre des données.
- Le terme « Ordinateur » a été proposé par le philosophe Jacques Perret en 1955.



a) **Processeur** (CPU = Central Processing Unit = Unité Centrale de Traitement) : est le composant essentiel de l'ordinateur pour exécuter les programmes (c'est l'endroit où sont effectués les traitement). Il est constitué de :
   a. UAL (Unité Arithmétique et Logique) dite aussi Unité de traitement : Elle est responsable sur tout ce qui est calcul.
   b. UC (Unité de contrôle) : Responsable sur le décodage des instructions et l'actionnement des commandes.

b) **Mémoire centrale** : Est un espace dans lequel résident les programmes à exécuter et les données à traiter.

c) **Unité d'entrée/sortie :** Est l'intermédiaire entre l'ordinateur et le monde externe.

d) **Horloge :** Responsable sur la génération des impulsions pour cadencer (donner un rythme régulier à) la vitesse du Processeur. Sa vitesse s'exprime en Hz/s.

2) Circulation de l'information dans un micro-ordinateur

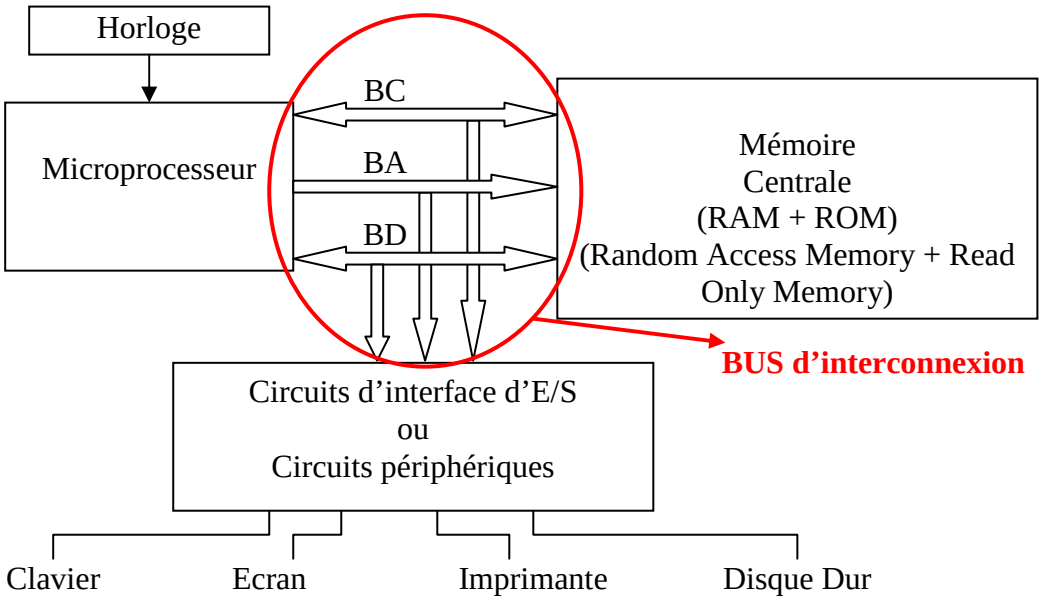
Def. Micro-ordinateur :

Est un ordinateur de faible encombrement construit au tour d'un microprocesseur qui constitue son Unité de Contrôle et de Traitement (UCT).

Microprocesseur :

C'est un processeur miniaturisé pour qu'il puisse tenir sur un seul circuit intégré.

Schéma fonctionnel d'un micro-ordinateur :



BUS : Ensemble de fils permettant d'interconnecter le microprocesseur, la mémoire et les périphériques.

BA :

- Bus d'Adresse.
- Unidirectionnel.
- Peut adresser $2^{(nbr\ fils)}$ de mots mémoire :
  2^1 → 2 cases mémoires adressables
  2^{16} → 65536 cases mémoire
  2^{20} → 1048576 cases mémoire

BD :

- Bus de Données.
- Bidirectionnel.
- Sert à la circulation des données (aussi des instructions) entre mémoire, microprocesseur et périphérique.
- 1 fil : Les données transportés sont à 1 seul bit : 0 ou 1
  8 fils : // // 8 bits : de 0 à 255 pour nombres non signés et de -128 à +127 pour nombres signés.
- Pour l'utiliser, il faut avoir d'abord le contrôle sur le bus.

BC : Bus de Contrôle (dit aussi Bus de Commande)

- Indique la nature de l'opération voulue par un processeur. Par exemple, Lecture/Ecriture depuis/dans la mémoire, entrée/sortie de/vers un périphérique.
- On trouve aussi dans le bus de contrôle une ou plusieurs lignes permettant aux circuits périphériques d'effectuer des demandes au µP. Ces lignes sont appelées « lignes d'interruption ».

- En général, le BC comporte 3 fils :

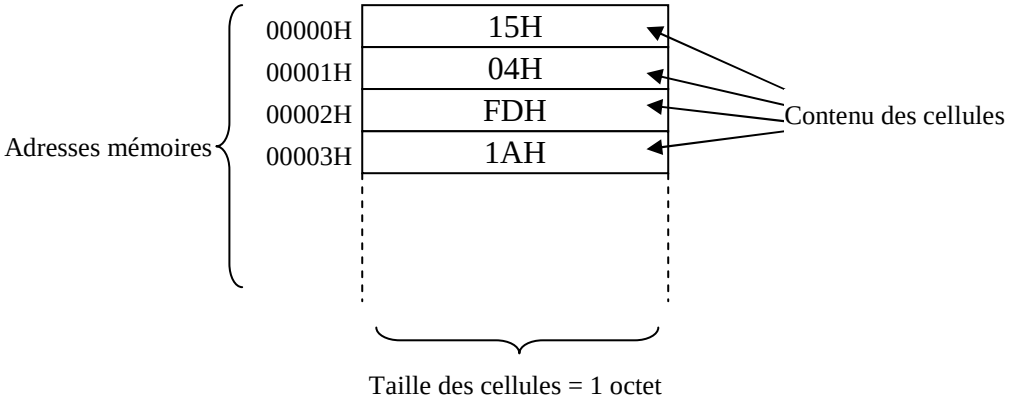
  a) $R/\overline{W}$ pour indiquer Lect ou Ecrit : 0 : W et 1 R

  b) CS (Chip Select) pour la sélection (activation) d'un composant afin d'autoriser R ou W. Certains composants sont sélectionnés par $\overline{CS}$

  c) OE (Output Enable) Permet l'activation des lignes de sortie : 1 mémoire est connectée sur les lignes de sortie, 0 déconnectée.

3) Structure de la mémoire centrale

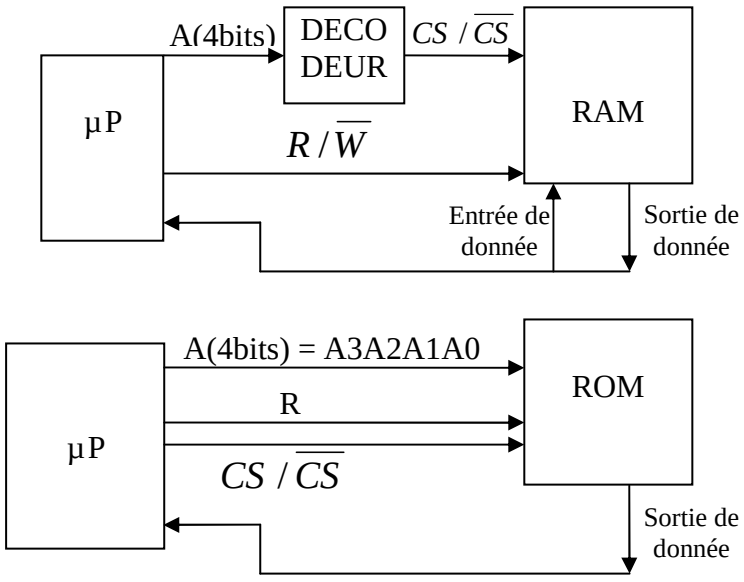
La mémoire peut être vue comme un ensemble de **cellules** ou de **cases** contenant. Chaque cellule mémoire est repérée par son indice (numéro d'ordre unique) qui représente son **adresse** :



Remarque :

Les cases mémoires peuvent être lues ou écrites par le µP(cas d'une RAM) ou bien seulement lues(cas d'une ROM).

Exemple :



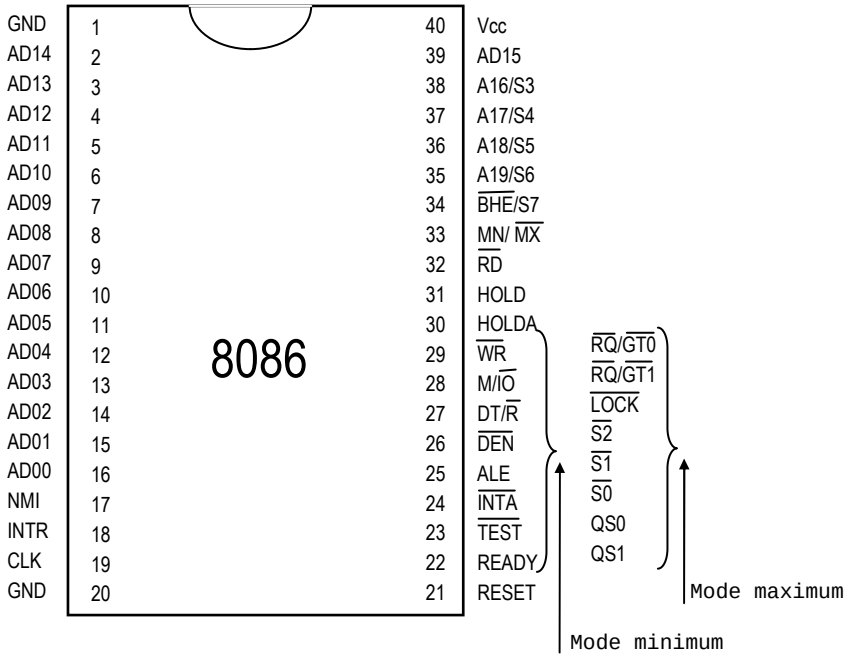
4) Description matérielle des µP (Cas du 8086)

Est un circuit intégré muni d'un nombre important de broches (pins). Par exemple, les µP Intel 8085, Intel 8086 et Zilog Z80 ont 40 broches, Motorola 68000 a 64 broches et Intel 80386 a 196 broches.

4.1) Vue externe du µP 8086 :

Le CPU INTEL 8086 est un µprocesseur à 16 bits qui contient environ 29 000 transistors. Il se caractérise par :

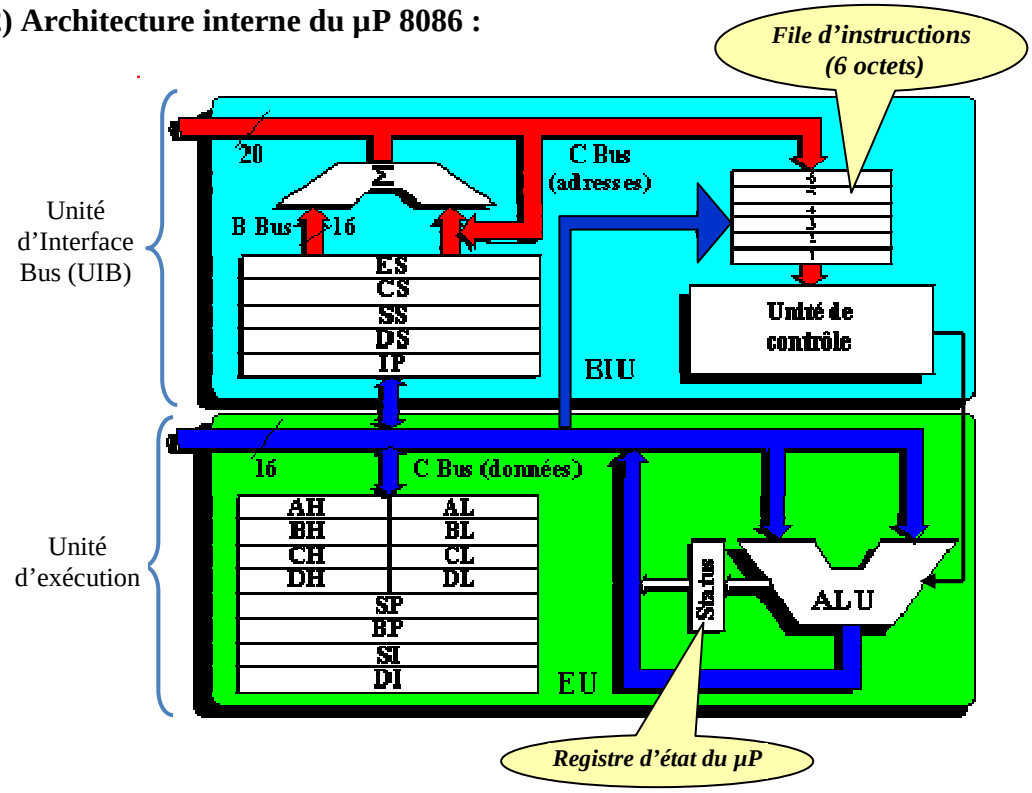
- Bus de données sur 16 bits
- Bus d'adresse sur 20 bits
- Taille des registres sur 16 bits
- 16 broches *AD0* jusqu'à *AD15* de données et d'adresses multiplexées, l'adresse et les données ne peuvent pas être envoyées en même temps;
- 04 broches *A16* jusqu'à *A19* de contrôle et d'adresse multiplexées ;
- 16 broches de contrôle;
- broche *VCC* +5V, broche *CLK* d'horloge de 5Mhz et 02 broches *GND* de masses.



<u>Remarque :</u>
Lignes de contrôle utilisées par la suite sont :

- ***R/<u>W</u>** : Lecture écriture ;*
- ***M/<u>IO</u>** : Pour cibler la MC ou un port d'E/S ;*
- ***INTR (Interrupt Request)** : La ligne de demandes d'interruption ;*
- ***<u>INTA</u> (Interrupt Acknowledge)** : Ligne d'acceptation de demandes d'interruption*

4.2) Architecture interne du µP 8086 :



Le 8086 comporte :

a) **Unité d'exécution:** Reçoit les instructions de la file d'instructions, les exécute et transmit le résultat à l'unité d'interface bus.

b) **Unité d'interface de bus:**
   i. Assure le contrôle des bus ;
   ii. Lit les instructions depuis la MC ;
   iii. Range les instructions dans la file d'instructions
   iv. Ecrit les résultats fournis par UE dans la MC.

c) **Les registres internes:** Un registre est un petit espace mémoire à l'intérieur du µP. Le 8086 dispose de 14 registres internes de 16 bits. Ces registres se classent en quatre catégories :

i. Les registres de données ou généraux:

Ce sont les registres utilisés pour la réalisation des opérations arithmétiques et logiques. Ils peuvent être utilisés comme des registres 8 bits (AH, AL, BH, ...) ou comme des registres 16 bits (AX, BX, CX et DX).

- AX (Accumulator register) : Registre accumulateur;
- BX (Base register) : Registre de base;
- CX (Counter register) : Registre compteur;
- DX (Data register) : Registre de données;

ii. Les registres pointeurs ou d'index:

Ce sont des registres qui peuvent être utilisé dans l'adressage mémoire.

- SI (Source Index register) : Registre d'index source;

- DI (Destination Index register) : Registre d'index destination;
- BP (Base Pointer register) : Registre pointeur de base pour pile;
- SP (Stack Pointer register) : Registre pointeur de pile.
- IP (Instruction Pointer) : registre pointeur d'instruction.

iii. Les registres segments :

Ce sont des registres utilisés lors de l'adressage mémoire. Ils contiennent les adresses de base des segments en MC :

- ES (Extra Segment) : Registre segment étendu. Il contient l'adresse de base du segment mémoire étendu qui est utilisé comme un segment de données supplémentaire ;
- CS (Code Segment) : Registre segment de code. Il contient l'adresse de base du segment mémoire qui contient les instructions du programme en cours d'exécution ;
- DS (Data Segment) : Registre segment de données. Il contient l'adresse de base du segment mémoire qui contient les données du programme en cours d'exécution ;
- SS (Stack Segment) : Registre segment de pile. Il contient l'adresse de base du segment mémoire dans lequel se trouve la pile.

iv. Le registre d'état du µP (PSW = Processor Status word) :

Il est sur 16 bits. Il comporte deux types de flags (drapeaux) :

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
				OF	**DF**	**IF**	**TF**	**SF**	**ZF**		**AF**		**PF**		**CF**

1. <u>Flags de condition :</u> Ils reflètent l'état du µP suite à une opérations arithmétique ou logique (ils donnent des informations sur le résultat) :

- <u>CF (Carry Flag):</u> pour signaler une retenue ou un emprunt. Exemple d'addition sur 08 bits : 1000 0000 + 1000 0000 positionne C à 1.
- <u>PF (Parity Flag):</u> 1 → Nombre de « 1 » contenus dans l'octet bas du résultat est pair.
  0 → impair
  .
- <u>AF (Auxiliary Flag):</u>
  1 → opération a produit une retenue ou un emprunt pour les 4 bits du poids faible.
- <u>ZF (Zero Flag):</u>
  1 → résultat de l'opération est nul.
- <u>SF (Sign Flag):</u> 1 → Résultat négatif, 0 → Résultat positif. Il reflète le bit de signe du résultat.
- <u>OF (Overflow Flag):</u> Indique que le résultat d'une opération signée a dépassé la capacité du contenant. Ce flag indique que le résultat n'est plus représentable.

2. ***Flags de contrôle :*** Ils permettent de contrôler quelques fonctionnalités du µP :
   - TF (Trap Flag):
     Pour exécuter un programme pas à pas. Ce bit est utilisé pour facilité la recherche des erreurs dans les programmes (débogage des programmes). Il n'existe pas des instructions permettant le modifier directement. On doit utiliser la Pile pour le modifier
   - IF (Interrupt Flag):
     1 → Autoriser les interruptions et donc le µP les accepte, c-à-d le µP réagit sur le signal INTR.
   - DF (Direction Flag): Fixe la direction des opérations répétitives. Ce flag est utilisé lors du traitement des chaînes de caractères.

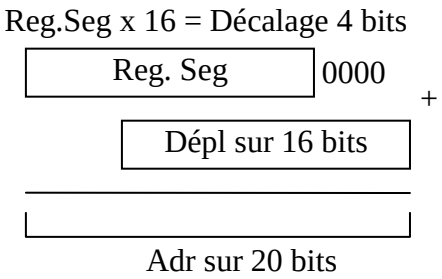
5) Segmentation de la mémoire (Adressage par segment)

5.1) Mode Réel :

Avec le 8086, le BA est sur 20 bits ➔ Espace mémoire adressable est de 2^{20} = 1 Mo.

Puisque les registres sont sur 16 bits alors comment faire pour calculer une adresse ?

L'idée est :



Donc, @physique = Reg. Seg * 16 + Dépl (sur 16 bits)

La taille des segments mémoires est 64 Ko.

Exemple :
Donner les valeurs de CS et IP pour l'adresse mémoire physique 157BBH ?

CS = 123A et IP = 341BH ; CS = 100BH et IP = 570BH, etc.

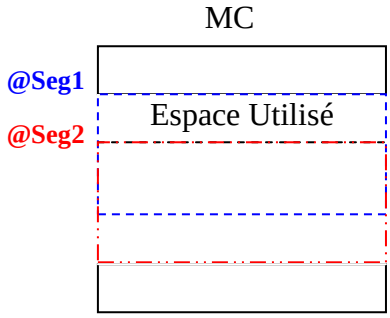
On remarque que pour la même adresse physique, CS et IP peuvent prendre des valeurs différentes à cause de la notation d'Intel ***RegSeg:Dépl***.

Remarque :
Chaque segment commence à une adresse de base de multiple 16

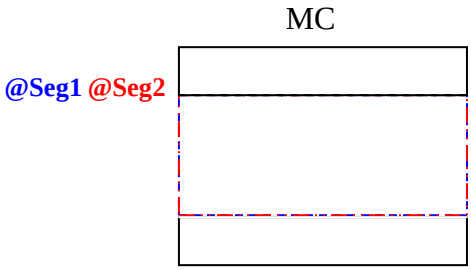
5.2) Chevauchement des segments :

Le but du chevauchement des segments est de permettre à un segment d'utiliser l'espace non utilisé par un autre segment :

a) Chevauchement partiel :



b) Chevauchement total

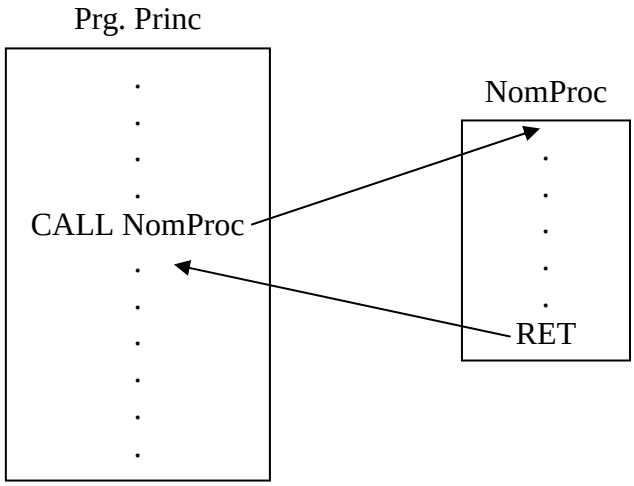


5.3) Pile

- Est un espace mémoire utilisé pour le stockage des données temporaires. Par exemple, pour garder la trace de retour à un programme après un appel d'une procédure, le passage des paramètres aux procédures ;
- L'accès à la pile est séquentiel suivant le principe LIFO (Last In First Out = Dernier entré premier sorti).
  SP : Pointe sur le sommet de la pile
- On ne peut Empiler/Dépiler qu'un seul élément à la fois :
  Empiler → : 1) SP = SP – 2
  2) Déposer (arg) à [SP].
  Dépiler → : 1) Récupérer (arg) depuis [SP]
  2) SP = SP + 2

5.3.1) Cas d'utilisation de la pile : Sauvegarde du contexte d'un programme :

Soit un programme qui appelle une procédure :



1) Appel Intra-segment : On sauvegarde que l'@ de l'instruction à partir de laquelle le µP reprend l'exécution du Prog. Princ. Donc,

a)sauver trace, puis charger @
SP ← SP – 2
[SP] ← IP ; @ de reprise
IP ← @ PremierInstProc ; chargement de l'@ de la 1ère instruction de la procédure

Une fois que µP termine l'exécution de la Proc, il exécuter RET qui permet la restauration de l'@ du retour (la reprise).

```
; Déclaration d'1 Proc Intra segment
MaProcIntraSeg      PROC  NEAR
        MOV AX, 1234H
        …
        RET
MaProcIntraSeg      ENDP
```

2) Appel Inter-segment : On sauvegarde l'@ ainsi que le registre segment de l'instruction à partir de laquelle le µP reprend l'exécution du Prog. Princ. Donc,

a)A l'appel
a.1) sauver trace
SP ← SP – 2
[SP] ← CS
SP ← SP – 2
[SP] ← IP
a.2) Charger @ seg et ip
CS ← @ SegmentProc
IP ← @ PremierInstProc

b)Au retour
b.1)Restaurer l'@ seg et ip du Prog. Princ.
IP ← [SP]
SP ← SP + 2
CS ← [SP]
SP ← SP + 2

6) Fonctionnement des µP (Cas du 8086)

Tout programme est composé d'un ensemble d'instructions. Son exécution revient à exécuter ses instructions, une par une, à l'intérieur du µP, en passant le cycle d'exécution de base composé de 3 étapes principales :

1) Rechercher l'instruction : Cette étape consiste à récupérer l'instruction à exécuter depuis l'adresse mémoire CS:IP et la placer dans le registre d'instruction. Ensuite, elle incrémente le registre IP pour qu'il pointe sur la prochaine instruction à exécuter ;
2) Décoder l'instruction : Cette étape consiste à identifier l'opérateur et les opérandes de l'instruction afin de sélectionner le circuit logique permettant la réalisation de l'instruction ;
3) Exécuter l'instruction : Cette étape se traduit par le placement des opérandes dans les tampons d'entrée et l'activation du circuit logique correspondant à l'opération.

Exemple :

En utilisant le cycle d'exécution de base, donner la trace d'exécution du programme suivant par le µP 8086 :

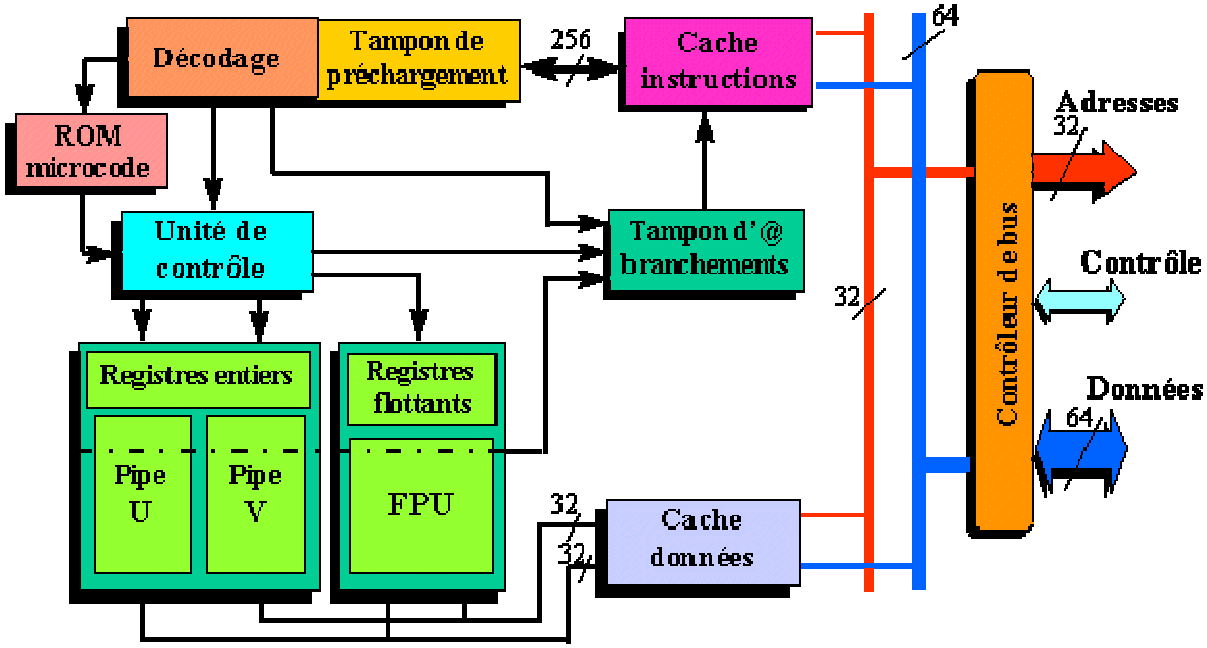
```
0400 :0100H    MOV    AL,    X
0400 :0102H    ADD    AL,    40H
```

Le contenu de l'adresse X = 15H

7) Amélioration des performances des µP

Problème d'augmentation de la vitesse d'horloge ➔ Chercher d'autres technologies pour augmenter les performances du µP.

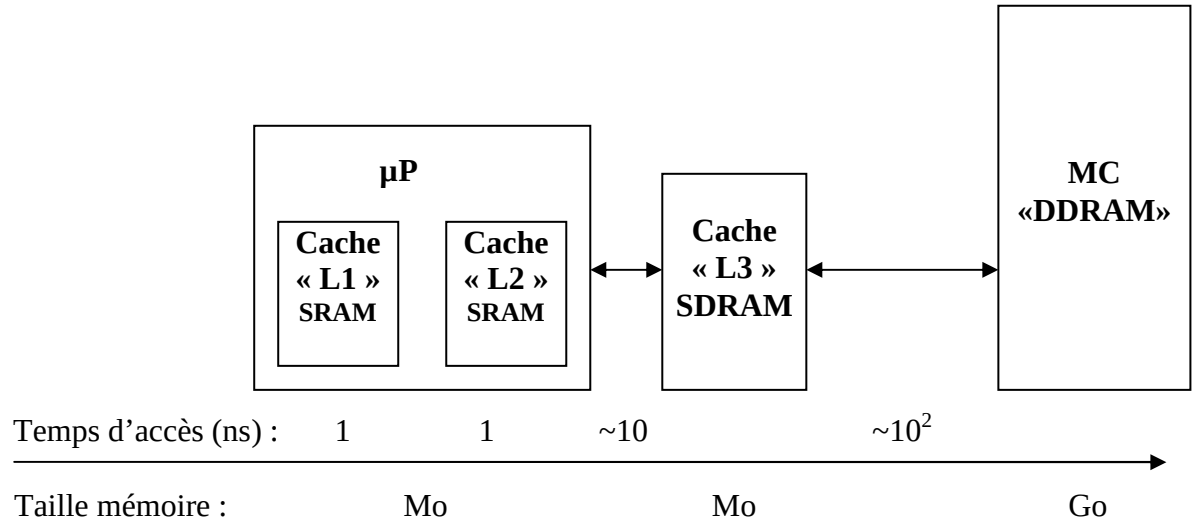
Diagramme de bloc du Pentium :



Tampon d'@ branchements: Permet le calcul d'une adresse de branchement avant l'exécution de l'instruction de branchement. L'adresse cible est ainsi disponible immédiatement. (Mode protégé).

ROM microcode: Les instructions comportant plus de 4 micro-opérations seront transférées dans ROM microcode afin d'établir leur ordonnancement.

7.1) Mémoires caches :



Remarque : SRAM est 8 à 16 fois plus rapide que DRAM mais 4 à 8 fois plus volumineuse

But :

1. Eviter de rechercher des données et des instructions dans la MC déjà recherchées précédemment en les conservant près du µP dans une mémoire cache. Très utile pour les instructions des boucles.

On distingue actuellement 3 niveaux de mémoire cache :

1. 2 L (L1 et L2) internes (dans le µP) qui sont des unités dont chacun contient séparément des
   instructions et des données ;
2. 1 L externe (à coté du µP);

Remarque :
Chaque cache mémoire est géré par un contrôleur de cache.

Exemple de fonctionnement :

1. Le processeur demande une instruction présente dans une boucle d'un programme en plaçant l'adresse de cette instruction sur le bus d'adresse ;
2. Le contrôleur de cache cherche si l'instruction est dans le cache en fonction de l'adresse. Si oui, l'instruction est immédiatement délivrée ;
3. Sinon, le contrôleur de cache entame un cycle de lecture en mémoire centrale ;
4. La mémoire centrale envoie l'instruction µP et le contrôleur de cache la copie au passage dans le cache pour qu'au prochain accès à cette même instruction, le contrôleur de cache l'enverra directement au µP sans démarrer un cycle de lecture en mémoire centrale.

7.2) Pipeline :

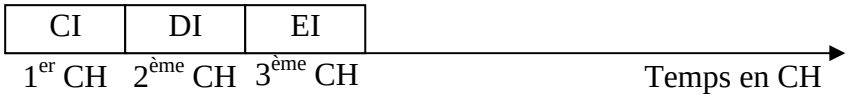
1 Pipeline est une technique de conception des processeurs où l'exécution de plusieurs instructions se chevauche à l'intérieur du processeur.

1) Principe du Pipeline :

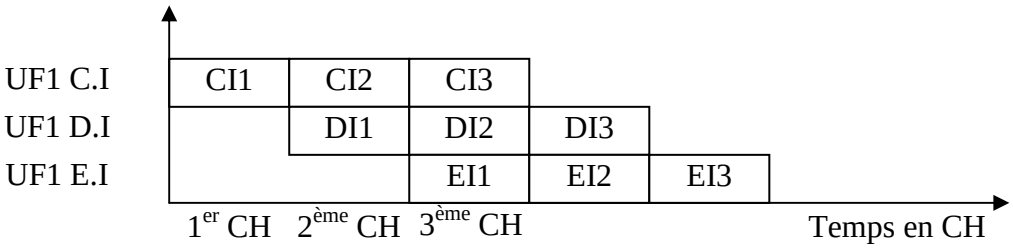
Pour comprendre le principe du pipeline, prenant l'exemple :
L'exécution d'une instruction par un µP passe par 3 étapes élémentaires :

1. Charger Instruction = CI (1 cycle d'horloge)
2. Décoder Instruction = DI (1 C.H)
3. Exécuter Instruction = EI (1 C.H)

Avec un µP sans pipeline, 1 instruction ne peut être exécutée avant la fin d'exécution de l'instruction qui la précède. En effet, pour exécuter N instructions, il faut 3*N C.H.



Par contre avec un µP avec pipeline, on utilise des unités fonctionnelles (UFs) où chacune d'elles exécute une étape. Pour notre exemple, on prend 3 UFs pour exécuter Les 3 étapes.



Remarque : Pipeline

- Permet, à partir d'un instant déterminé, à exécuter 1 instruction par cycle d'horloge. Dans notre exemple, à partir du 3ème Cycle Horloge.
- Amélioration du temps d'exécution. Pour notre exemple, N instructions sont exécutées en N+2 CH. En effet, on calcul 3 fois plus rapide / au µP sans pipeline.
- Il possible d'augmenter le nombre d'étages du pipeline (exemple : 2 UF1 C.I)

Exemple :

- Pentium4 a un pipeline de 20 étages.

2) Problème posé par les Pipeline :

L'interdépendance des instructions :

Exemple d'interdépendance des données:
ADD R1, R2, R3
MOV R4, R1
➔ R4 va contenir la val de R1 avant ADD et non pas après.

7.3) Format d'instruction (famille d'appartenance des µP):

1. *CISC (Complex Instruction Set Computer = µP à jeu d'instruction étendu)* :
   Un µP possédant un jeu d'instructions comprenant de très nombreuses instructions mixées à des modes d'adressages complexes.

   µP de cette famille sont les x86.

   Avantage :
   - Permet la microprogrammation
   - Permet l'utilisation des instructions plus complexes.

   Inconvénient :
   - µP plus compliqué à accélérer ➔ introduction des pipelines pose des pbs.
   - µP CISC est globalement plus complexe qu'1 µP RISC
   - Utilisation d'un microprogramme réalisant l'exécution des instructions.

2. *RISC (Reduced Instruction-Set Computer = µP à jeu d'instruction réduit)*:
   On a remarqué que seul un jeu d'instruction était principalement utilisé dans les programmes. C'est pourquoi l'architecture RISC fait le choix de limiter le jeu d'instructions à seulement quelques unes, imposant, en contrepartie, à toutes les instructions, un nombre identique de cycles pour s'exécuter.

   µP Pentium a une architecture hybride (CISC émulé par RISC)

   Avantage :
   - Amélioration de la performance des µP.
   - µP RISC est plus simple et souvent petit, ce qui permet de lui ajouter des mémoires caches.
   - La réalisation des instructions peut se faire par des séquenceurs câblés.