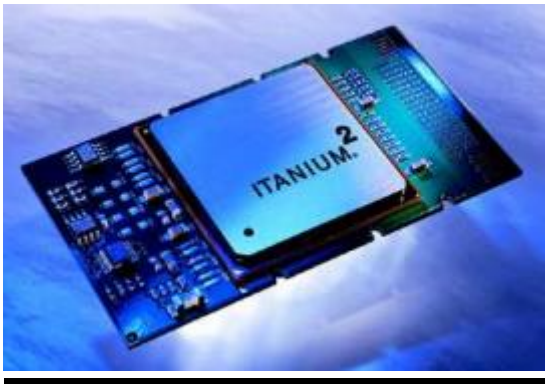
AU CŒUR DU PROCESSEUR



Le **processeur central** ou **unité centrale de traitement** (UCT/CPU) est un circuit intégré complexe caractérisé par une très grande intégration et doté de facultés d'interprétation et d'exécution des instructions d'un programme ; il prend aussi en compte les informations extérieures au système et assure leur traitement. Véritable cerveau de l'ordi, le CPU est intimement associé à la mémoire centrale où sont stockées les infos à traiter.

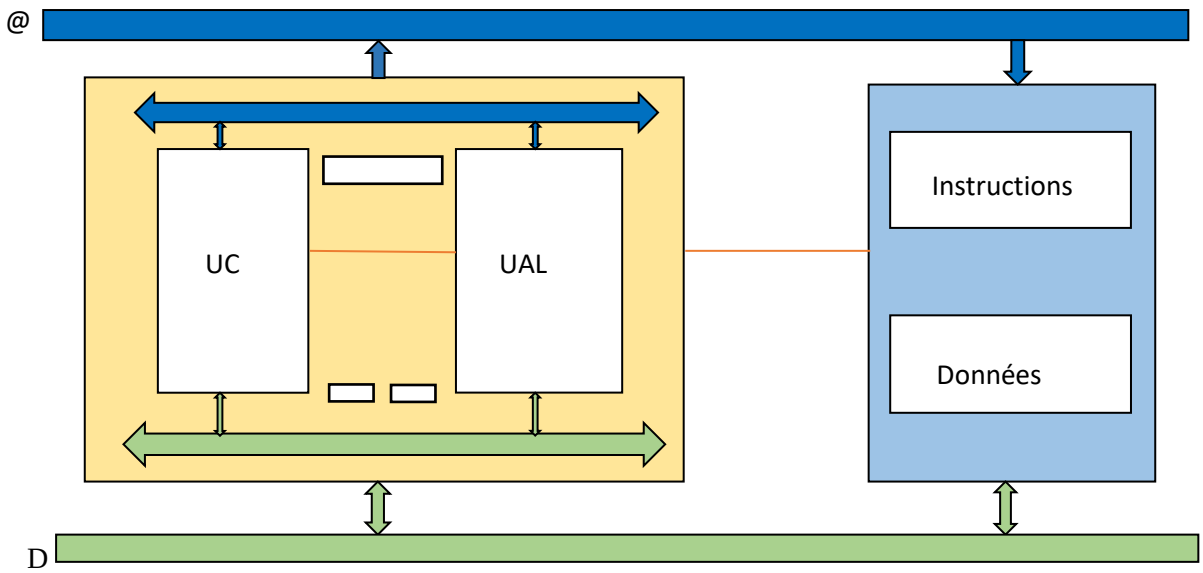
I. ARCHITECTURE DE BASE D'UN PROCESSEUR

1. Description

Un microprocesseur est construit autour de deux éléments principaux :

- Une unité de commande et de contrôle (**UC**)
- Une unité de traitement (**UAL**)

associés à des **registres** et à la **mémoire cache** où le CPU range temporairement, sans accéder à la mémoire, données, instructions ou adresses. Ces éléments sont reliés entre eux par des **bus internes** permettant les échanges d'informations ; des **bus externes** relient l'ensemble à la mémoire centrale et à la vidéo.



2. Fonctionnement général

L'unité de commande pour une opération de lecture, va à la MC à une adresse indiquée, chercher une instruction enregistrée sous forme binaire, la décode à fin de déterminer l'opération en question, ce qui lui permet de générer les signaux nécessaires à l'UAL pour l'exécution de l'instruction. Les données à traiter seront aussi cherchées en mémoire et transférées directement à l'UAL.

L'unité de commande envoie une adresse et une commande à la MC par rapport à une information (en lecture ou en écriture), celle-ci lui/y est transférée selon la commande (impulsion de lecture : Read ou celle d'écriture : Write). Dans le cas de la lecture d'une instruction, elle décode celle-ci pour déterminer l'opération demandée, qui lui permet de générer les signaux nécessaires à l'UAL pour exécuter l'instruction. Les données à traiter sont cherchées en mémoire et transférées à l'UAL. L'unité de commande déclenche, contrôle et synchronise donc l'activité de l'UAL.

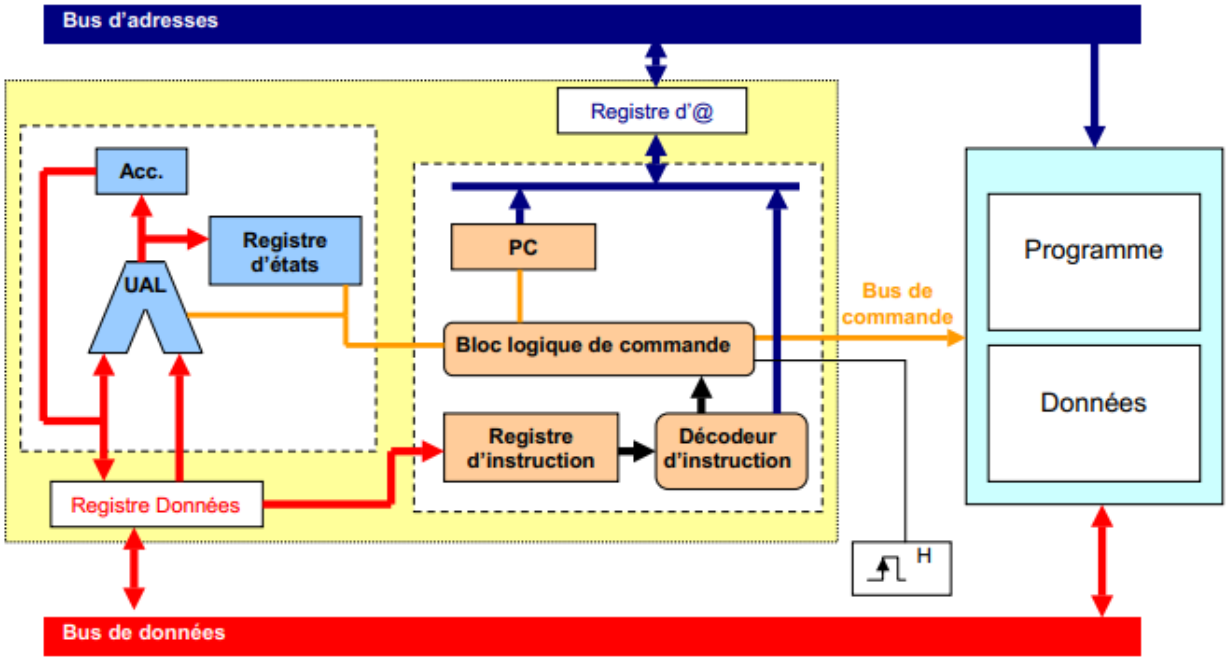


Schéma fonctionnel

II. L'unité de commande (UC)

1. Définition

L'unité de commande est un ensemble de dispositifs coordonnant le fonctionnement de l'ordinateur afin de lui faire exécuter la suite d'opérations spécifiées dans les instructions du programme.

Elle intervient dans les cycles (2 ou 3) de fonctionnement du processeur (Fetch-decode et Execute ou Fetch, Decode et Execute).

2. Principaux dispositifs de l'unité de commande selon le cycle.

➢ Traitement d'une instruction en deux phases ou cycles :

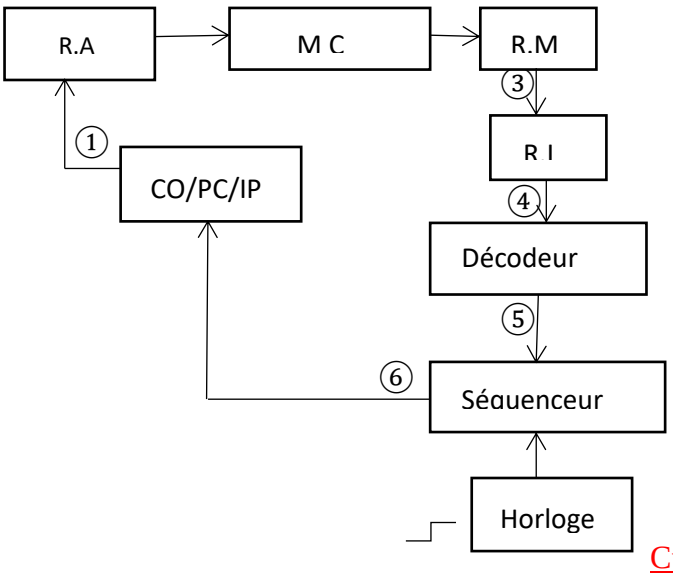
❖ Pour le cycle de recherche et de décodage de l'instruction, on a :

- Le **compteur ordinal (CO) / PC** (program counter) / **IP** (instruction pointer) : registre qui reçoit l'adresse de l'instruction à chercher.
- Le **registre instruction (RI)**: reçoit l'instruction à exécuter
- Le **décodeur** de code opération : détermine l'opération à effectuer
- Le **séquenceur ou bloc logique de commande** : génère les signaux de commande.
- L'**horloge** : synchronise les actions de l'unité centrale par ses impulsions.

❖ Pour le cycle d'exécution (qui a lieu après celui de recherche et décodage), on a :

- Le séquenceur,
- L'UAL : dispositif où s'effectue l'opération spécifiée dans l'instruction cherchée.
- Le registre tampon de l'UAL (RTUAL) : stocke temporairement l'un des deux opérandes d'une instruction arithmétique.
- L'accumulateur : registre qui stocke les opérandes et les résultats des opérations de l'UAL.
- Le registre d'état (PSW) : Il est généralement composé de 8 bits à considérer individuellement. Chacun de ces bits est un indicateur dont l'état dépend du résultat de la dernière opération effectuée par l'UAL. On les appelle *indicateur d'état* ou *flag* ou *drapeaux*. Dans un programme le résultat du test de leur état conditionne souvent le déroulement de la suite du programme. On peut citer par exemple les indicateurs de :
  - retenue **(carry : C)**
  - retenue intermédiaire **(Auxiliary-Carry : AC)**
  - signe **(Sign : S)**
  - débordement **(overflow : OV ou V)**
  - zéro **(Z)**
  - parité **(Parity : P)**

Circulation des informations pendant les deux cycles :



① : Xfert de la nouvelle adresse du CO au RA

② : Transfert de l'instruction cherchée vers RM

③ : Transfert de l'instruction dans RI

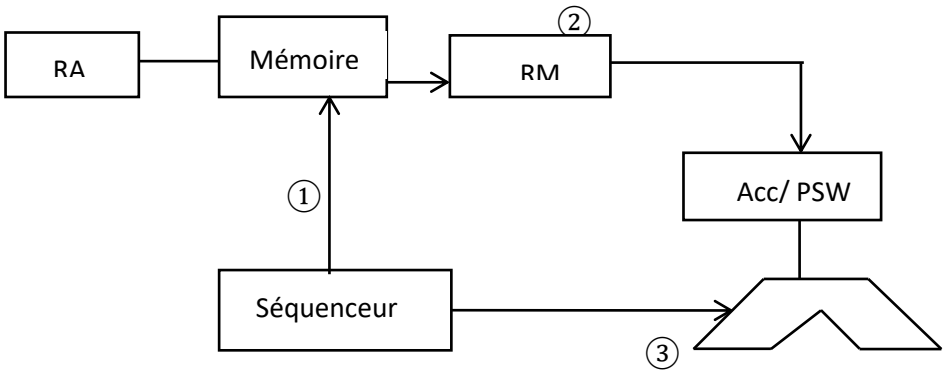
④ : Transmission du code opº vers le décodeur et l'adresse de l'opérande vers RA

⑤ : Xmission du type d'opº au séquenceur

⑥ : Incrémentation du CO pour le cycle suivant.

Note : X = trans.

Cycle Fetch and decode



①: Envoie de Read pour lire l'opérande à l'adresse dans RA, résultat placé dans RM

② : Xfert du contenu de RM vers l'UAL (accu ou reg).

③ : Réalisation de l'opº sous le ctrl du séquenceur.

Cycle Execute

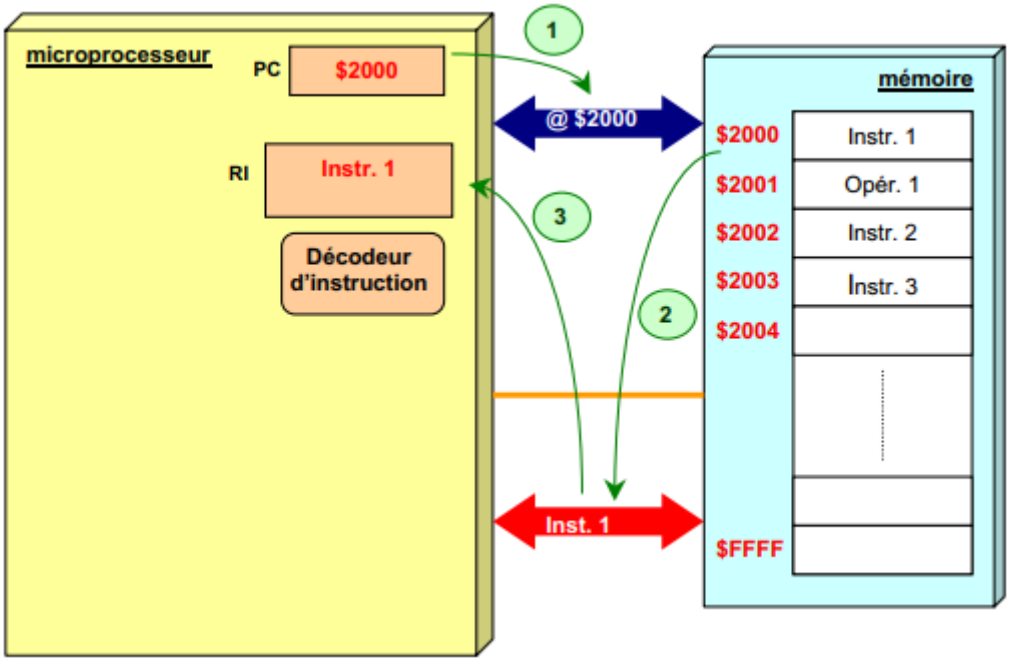
Traitement d'une instruction en trois phases ou cycles :

- **Phase 1:** *Recherche de l'instruction à traiter*

1. Le PC contient l'adresse de l'instruction suivante du programme. Cette valeur est placée sur le bus d'adresses par l'unité de commande qui émet un ordre de lecture.

2. 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.

3. L'instruction est stockée dans le registre instruction du processeur.



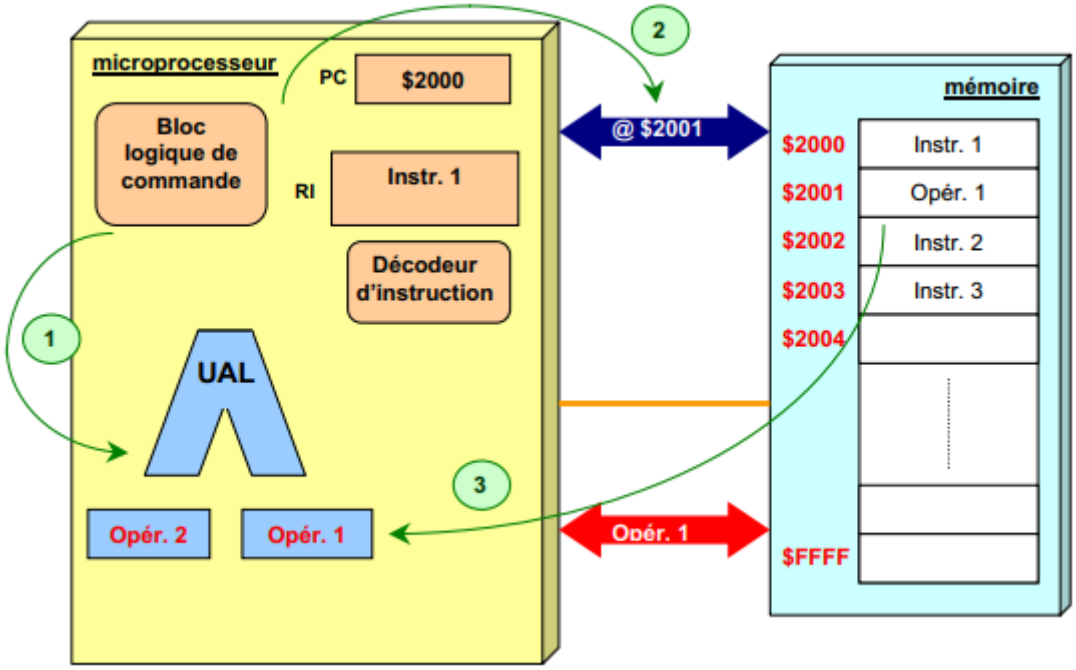
- **Phase 2:** *Décodage de l'instruction et recherche de l'opérande*

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.

1. L'unité de commande transforme l'instruction en une suite de commandes élémentaires nécessaires au traitement de l'instruction.

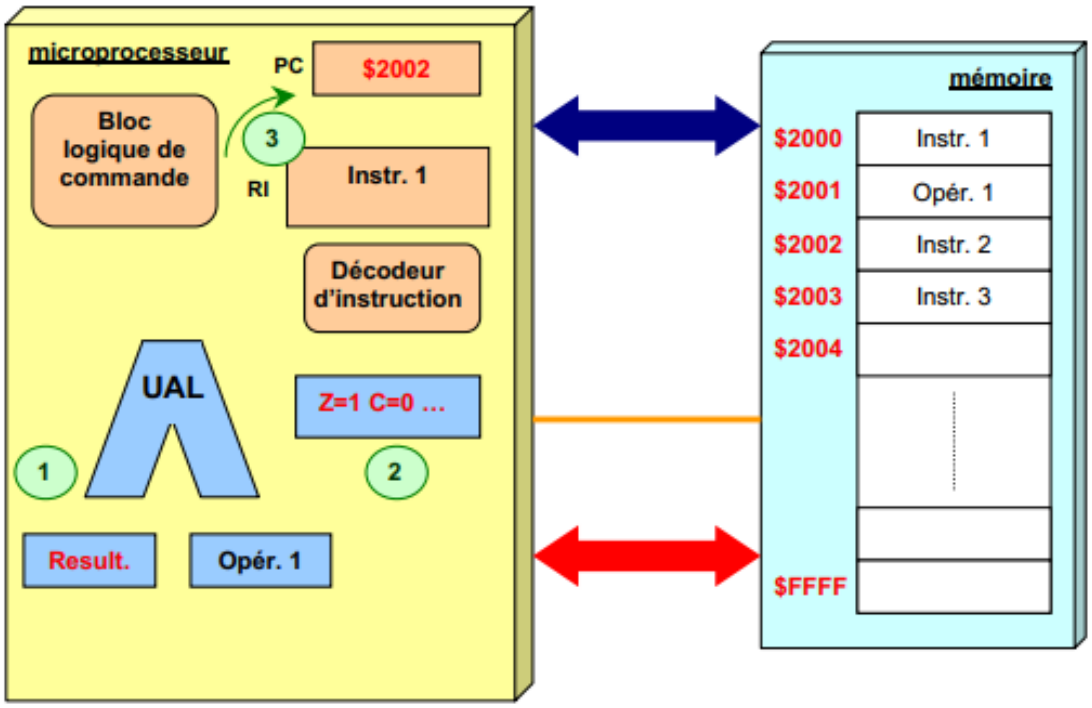
2. Si l'instruction nécessite une donnée en provenance de la mémoire, l'unité de commande récupère sa valeur sur le bus de données.

3. L'opérande est stocké dans un registre.



- **Phase 3 :** *Exécution de l'instruction*

1. Le microprogramme réalisant l'instruction est exécuté.
2. Les drapeaux sont positionnés (*registre d'état*).
3. L'unité de commande positionne le PC pour l'instruction suivante.



3. La synchronisation

Des circuits de l'unité de commande synchronisent et contrôlent toutes les opérations.

Le cycle de base ou cycle machine, durée élémentaire régissant le fonctionnement de la machine, est définie par les signaux périodiques générés par l'horloge.

Le cycle instruction, composé des cycles recherche et exécution peut s'étendre sur plusieurs cycles machine.

Le cycle CPU est le temps d'exécution de l'instruction la plus courte ou la durée d'une action élémentaire provoquant un changement d'état.

La vitesse de l'ordinateur dépend de sa fréquence d'horloge, de son cycle, et de la structure de la mémoire, de même que du temps d'accès en cache.

4. Le séquenceur

C'est un automate gérant les signaux de commande nécessaires pour activer et contrôler les unités participant à l'exécution d'une instruction donnée. Ces signaux sont distribués aux différents points de commande des organes concernés.

Il est câblé ou microprogrammé ; dans une mémoire ROM ou EPROM. Dans le premier cas, c'est un circuit séquentiel formé de sous-circuits dédiés aux instructions microprogramées ; alors que dans le deuxième, c'est une mémoire de microprogrammation stockant de microinstructions (qui génèrent des signaux de commande).

5. Jeu d'instructions

Chaque machine a son jeu d'instructions de base. Pour ce critère, il existe deux architectures : RISC et CISC ;

❖ **Architecture RISC** (Reduced Instructions Set Computer)

Elle préconise un petit nombre d'instructions élémentaires, faciles à réaliser dans le matériel et d'exécution rapide. Son séquenceur est câblé et le compilateur exploite à fond les caractéristiques de la machine.

Inconvénients :

- Programmation difficile,
- pas de fonctions supplémentaires câblées

Avantages:

- Compilateur plus puissant,
- coût réduit,
- instructions exécutables en un cycle d'horloge d'où une exécution des programmes plus rapide,
- traitement en parallèle des instructions.

❖ **Architecture CISC** (Complex Instructions Set Computer) :

Elle est utilisée par tous les processeurs type X86 (Intel, AMD, VIA …). Des instructions complexes sont directement câblées (imprimées) sur la puce, d'où une rapidité d'exécution. Le séquenceur est microprogrammé.

Inconvénients :

- Coût élevé dû à la fabrication (câblage des instructions),
- instructions couvrant plus d'un cycle d'horloge, d'où lenteur.

Avantages:

- Instructions riches et composées comme : la racine carrée, la multiplication en virgule fixe ou flottante simple ou double précision ;

Comparaison des deux architectures :

Architecture RISC	Architecture CISC
• instructions simples ne prenant qu'un seul cycle ; • instructions au format fixe ; • décodeur simple (câblé) ; • beaucoup de registres ; • seules les instructions MOVE et STORE ont accès à la mémoire ; • peu de modes d'adressage ; • compilateur complexe.	• instructions complexes prenant plusieurs cycles ; • instructions au format variable ; • décodeur complexe (microcode) ; • peu de registres ; • toutes les instructions sont susceptibles d'accéder à la mémoire ; • beaucoup de modes d'adressage ; • compilateur simple.

Remarque :

Les instructions courantes sont par groupes :

- Transfert de données (LODE, MOVE, STORE) ;
- Opérations arithmétiques (les 4 opérations en virgules fixe ou flottante simple ou double précision) ;
- Opérations arithmétiques et logiques (AND, OR, NOT, XOR, …)
- Contrôle de séquence (branchements impératifs et conditionnels, appels de procédure …),
- I/O (E/S) (READ, WRITE, PRINT, etc.)
- Manipulations diverses (décalages, conversions de format, incrémentation de registres, gestion de la mémoire …).

6. Les registres du CPU

Ils déterminent l'architecture du CPU et influence la programmation. On a :

- Le registre PC : automatiquement incrémenté après chaque utilisation, d'où une exécution séquentiel du programme. Sa taille est fonction de celle de la mémoire centrale : elle doit permettre d'adresser la totalité de la mémoire. Le programmeur n'y a pas accès.
- Le registre instruction : le processeur y place l'instruction cherchée ou à stockée en mémoire. Sa taille est égale à celle du mot-mémoire. Les bits du code opération sont envoyés au décodeur. Le programmeur n'y a pas accès.
- L'accumulateur : Il contient l'un des opérandes avant exécution et le résultat après pour les opérations arithmétiques et logiques ; pour les opérations d'E/S, il sert de tampon. Il a la taille du mot-mémoire et possède une extension, le registre Q qui permet de doubler sa taille, ou qui contient le résultat, le quotient ou le dividende d'une multiplication. Le programmeur peut y accéder. ex : EAX, AX dans les X86
- Les registres généraux ou banalisés ou à usage général (RUG) (general purpuse registers) ou bloc-notes (scratchpad) : ils évitent les accès fréquents à la mémoire en conservant les résultats intermédiaires et les informations fréquemment utilisées par un programme.

  Ex : cas des IA32 : EAX, EBX, ECX, EDX ; BP, SP, SI, DI. Certains ont quand même un rôle spécifique: EAX est utilisé pour l'addition et la multiplication, ECX comme compteur de boucle, EBP pour les langages de haut niveau.
- Le registre d'état (PSW=Program Status Word) ou registre condition: Il utilise les différents drapeaux (bits) pour indiquer l'état d'une condition particulière dans le CPU.
- Le registre pointeur de pile SP (Stack Pointer) : Utilisé pour simuler une pile en mémoire centrale, il fonctionne comme un registre d'adresse : il indique à tout moment l'adresse du sommet de la pile. Ex : *ESP des IA-32*
- Les registres d'index ou d'indice (XR), de base ; utilisés pour le calcul des adresses effectives, ils sont utilisés pour les décalages, les opérations en virgule flottante.

Ex : *ESI et EDI sont des registres d'index pour les transferts mémoire à haute vitesse*

7. Codage des instructions et Adressage des opérandes.

Une instruction contient toutes les informations nécessaires à la machine pour exécuter une opération élémentaire. Les instructions et les opérandes sont stockés en mémoire centrale. La taille totale d'une instruction dépend de son type et aussi de celui d'opérande. Caque instruction est toujours codée en un nombre entier d'octets, afin de faciliter son décodage.

Une instruction est composée de deux champs :

- Le champ code opération, qui indique au processeur le type d'action demandée (addition, test, comparaison…) ;
- Le champ adresse/opérande qui contient la donnée ou la référence en mémoire de la donnée ou de l'instruction suivante.

Champ code opération	Champ code opérande

Ex : A1 0110

Nota : le champ adresse peut contenir une ou plusieurs adresses : on parle d'instructions à n adresses et donc d'une machine à n adresses (n=1,..,4) ; les machines à 0 adresse utilise la pile ou mémoire LIFO et celle à une adresse utilise l'accumulateur.

Ils existent plusieurs modes d'adressage selon le contenu du champ d'adresse de l'instruction ; et selon le mode, une instruction peut être codée sur 1, 2, 3, ou 4 bytes. Les modes d'adressages les plus courants sont :

- ***Adressage implicite :***
  l'instruction contient seulement le code opération : c'est une opération sans opérande ou portant sur un registre. Ex : INC (incrémenter l'accumulateur, EAX par exemple)
- ***Adressage immédiat :***
  le champ adresse contient l'opérande, une constante. Ex : ADD 5 (ajouter 5 EAX)
- ***Adressage direct :***
  Le champ adresse contient l'adresse de la donnée. Ex LOAD 120 : (transférer dans EAX la valeur à l'adresse 120h)
- ***Adressage indirect :***
  le champ d'adresse contient l'adresse où se trouve l'adresse effective.
- ***Adressage indexé :***
  adresse effective = adresse du champ d'adresse + contenu du registre d'index
- ***Adressage basé :***
  idem au précédent mais le registre d'index étant remplacé par le registre de base.
- ***Adressage relatif :***
  idem au précédent mais utilise le PC comme adresse de base.

Remarque : On peut combiner les modes d'adressage, par exemple l'indexé et l'indirect.

Exemples d'adressage : Effet sur l'accumulateur, de l'opération de transfert sous différents modes

Soit une machine disposant d'un accumulateur et dont les cellules mémoires contiennent les données a, b, c, epsi, oméga, ... comme indiqué. IMM (immédiat), I (indirect), XR1 (registre d'index n°1), B1 (registre de base n°1).

Etats des registres et des mémoires				
Adresses	Contenu		Adresses	Contenu
100	a		300	alpha
101	b		301	bêta
102	c		302	gamma
103	d		XR1	1
200	300		B1	100
201	302		B2	200

Dans la table ci-dessous, on a les exemples d'opérations sur l'accumulateur ; les modes d'adressages sont indiqués après le champ adresse, séparés par des virgules.

Instruction	**Effet sur l'Acc**
MOVE 100	a
MOVE 100, IMM	100
MOVE 200, I	alpha
MOVE 200, XR1	302
MOVE 200, XR1, I	gamma
MOVE 200, I, XR1	bêta
MOVE, 3, B1	d
MOVE, 1, B2	302

III. Unité arithmétique et logique (UAL)

Elle réalise une grande variété d'opérations (complémentation, mise à zéro, addition, …, ET, OU, …, XOR). Elle est asservie à l'unité de commande. Elle contient tous les circuits capables d'effectuer les opérations élémentaires qui sont à la base de tout algorithme. Les ordinateurs scientifiques disposent d'une gamme d'opérations en virgule flottante.

IV. Améliorations de l'architecture de base

L'ensemble des améliorations des microprocesseurs visent à diminuer le temps d'exécution du programme.

La première idée qui vient à l'esprit est d'augmenter tout simplement la fréquence de l'horloge du microprocesseur. Mais l'accélération des fréquences provoque un surcroît de consommation ce qui entraîne une élévation de température. On est alors amené à équiper les processeurs de systèmes de refroidissement ou à diminuer la tension d'alimentation.

Une autre possibilité d'augmenter la puissance de traitement d'un microprocesseur est de diminuer le nombre moyen de cycles d'horloge nécessaire à l'exécution d'une instruction. Dans le cas d'une programmation en langage de haut niveau, cette amélioration peut se faire en optimisant le compilateur. Il faut qu'il soit capable de sélectionner les séquences d'instructions minimisant le nombre moyen de cycles par instructions. Une autre solution est d'utiliser une architecture de microprocesseur qui réduise le nombre de cycles par instruction.

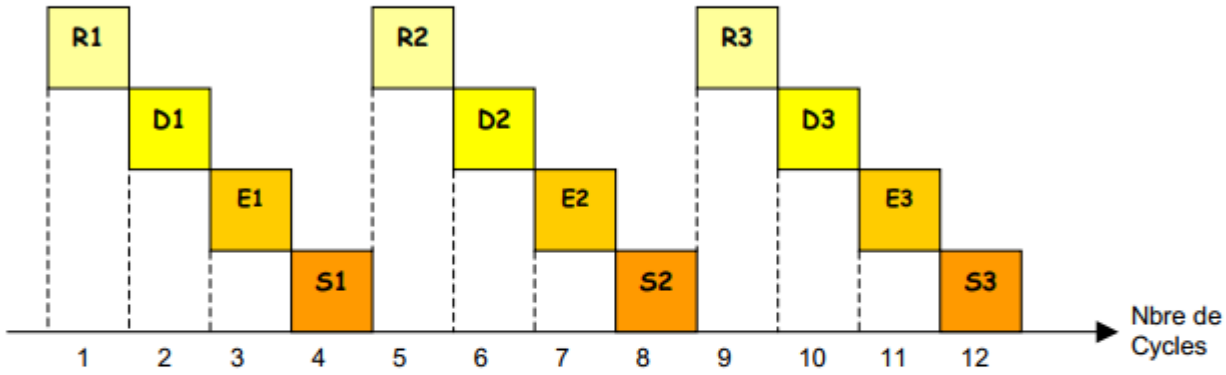
1. Architecture pipeline

a) Principe

L'exécution d'une instruction est décomposée en une succession d'étapes et chaque étape correspond à l'utilisation d'une des fonctions du microprocesseur. Lorsqu'une instruction se trouve dans l'une des étapes, les composants associés aux autres étapes ne sont pas utilisés. Le fonctionnement d'un microprocesseur simple n'est donc pas efficace.
L'architecture pipeline permet d'améliorer l'efficacité du microprocesseur. En effet, lorsque la première étape de l'exécution d'une instruction est achevée, l'instruction entre dans la seconde étape de son exécution et la première phase de l'exécution de l'instruction suivante débute. Il peut donc y avoir une instruction en cours d'exécution dans chacune des étapes et chacun des composants du microprocesseur peut être utilisé à chaque cycle d'horloge. L'efficacité est maximale. Le temps d'exécution d'une instruction n'est pas réduit mais le débit d'exécution des instructions est considérablement augmenté. Une machine pipeline se caractérise par le nombre d'étapes utilisées pour l'exécution d'une instruction, on appelle aussi ce nombre d'étapes le nombre d'étages du pipeline.

Exemple de l'exécution en 4 phases d'une instruction :

Modèle classique :



Modèle pipeliné :

b) Gain de performance

Dans cette structure, la machine débute l'exécution d'une instruction à chaque cycle et le pipeline est pleinement occupé à partir du quatrième cycle. Le gain obtenu dépend donc du nombre d'étages du pipeline. En effet, pour exécuter **n** instructions, en supposant que chaque instruction s'exécute en **k** cycles d'horloge, il faut :

- **n.k** cycles d'horloge pour une exécution séquentielle.
- **k** cycles d'horloge pour exécuter la première instruction puis **n-1** cycles pour les **n-1** instructions suivantes si on utilise un pipeline de **k** étages

Le gain obtenu est donc de :

$$G=\frac{n.k}{k+n-1}$$

Donc lorsque le nombre **n** d'instructions à exécuter est grand par rapport à **k**, on peut admettre qu'on
divise le temps d'exécution par **k**.

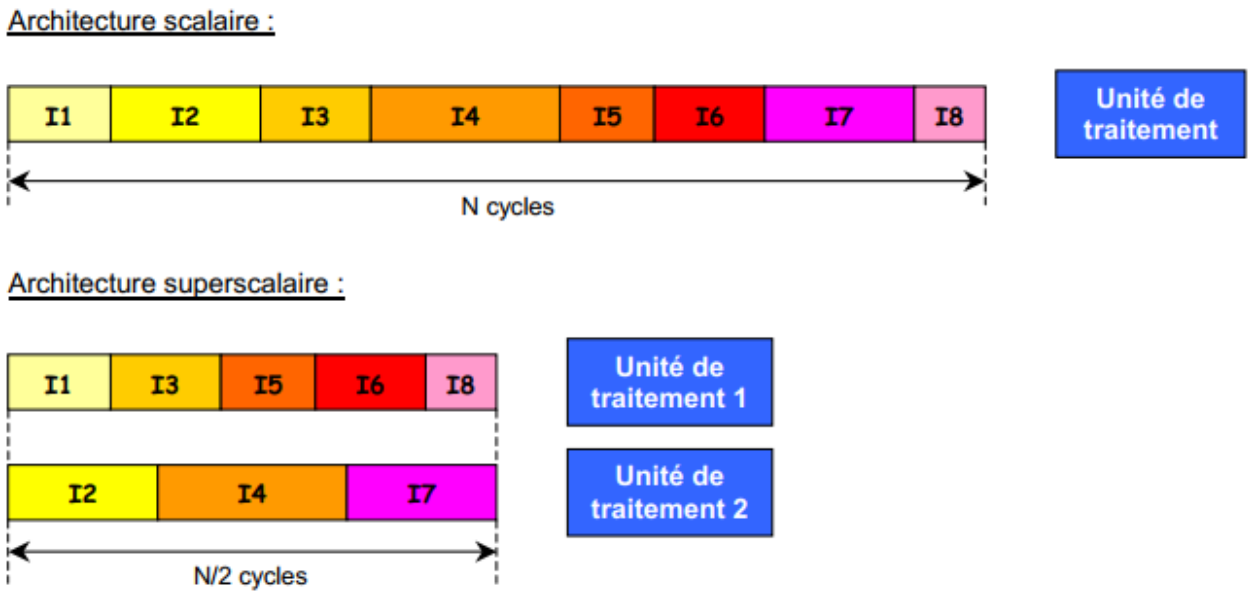
Remarques :

Le temps de traitement dans chaque unité doit être à peu près égal sinon les unités rapides doivent
attendre les unités lentes.

Exemples :
L'Athlon d'AMD comprend un pipeline de 11 étages.
Les Pentium 2, 3 et 4 d'Intel comprennent respectivement un pipeline de 12, 10 et 20 étages.

2. Architecture superscalaire

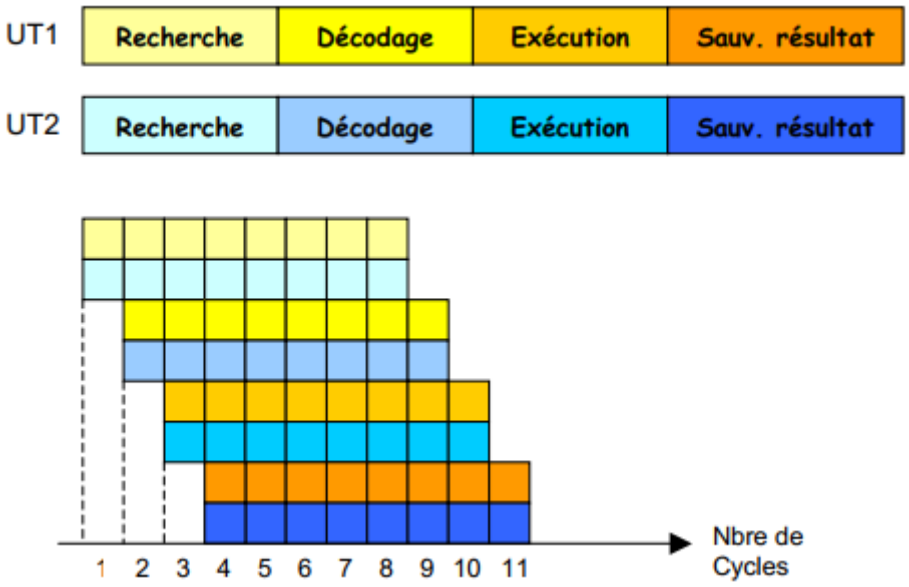
Une autre façon de gagner en performance est d'exécuter plusieurs instructions en même temps. L'approche superscalaire consiste à doter le microprocesseur de plusieurs unités de traitement travaillant en parallèle. Les instructions sont alors réparties entre les différentes unités d'exécution. Il faut donc pouvoir soutenir un flot important d'instructions et pour cela disposer d'un cache performant.



Remarque :
C'est le type d'architecture mise en œuvre dans les premiers Pentium d'Intel apparus en 1993.

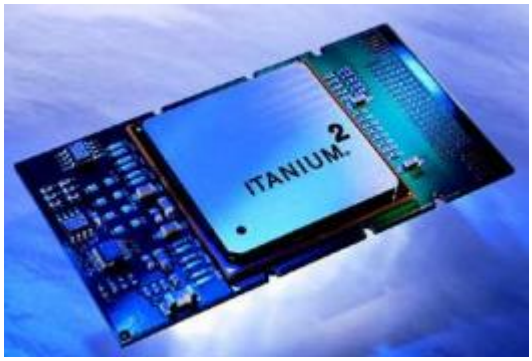
3. Architecture pipeline et superscalaire

Le principe est de d'exécuter les instructions de façon pipelinée dans chacune des unités de traitement travaillant en parallèle.


UT1
Recherche
Décodage
Exécution
Sauv. résultat
UT2
Recherche
Décodage
Exécution
Sauv. résultat
1 2 3 4 5 6 7 8 9 10 11
Nbre de
Cycles

Evolution de la technologie des processeurs

Processeur	Nb de transistors	Année
1ère CPU Intel	2300	1971 (c
8086	29000	1980
80286	130000	1982
80386DX4	280000	1985
80386SX	275000	1987
486DX4	1,2 millions	1989
486SX	1,185 millions	1991
486DX2	1,2 millions	1992
Pentium	3,2 millions	1993
486DX4	1,6 millions	1994
Pentium Pro	5 millions	1995
Pentium MMX	4,5 millions	1996
Pentium II	7,5 millions	1998
AMD K6	8,8 millions	1997
Celeron	7,5 millions	1998
AMD Athlon (K7)	22 millions	1999
Pentium 4	42 millions	2001
Pentium Extreme	169 millions	2004
Core 2 Duo	291 millions	2006
Core 2 Quad	582 millions	2006
Dual Core Itanium 2	1,7 milliard	2007



V. Processeurs spéciaux

1. Le microcontrôleur

Ce sont des systèmes minimum sur une seule puce. Ils contiennent un CPU, de la RAM, de la ROM et des ports d'Entrée/Sorties (parallèles, séries, etc..). Ils comportent aussi des fonctions spécifiques comme des compteurs programmables pour effectuer des mesures de durées, des CAN voir des CNA pour s'insérer au sein de chaînes d'acquisition, des interfaces pour réseaux de terrain, etc ...

Il est adapté pour répondre au mieux aux besoins des applications embarquées (appareil électroménagers, chaîne d'acquisition, lecteur carte à puce, etc...). Il est par contre généralement moins puissant en termes de rapidité, de taille de données traitables ou de taille de mémoire adressable qu'un microprocesseur.

2. Le processeur de signal

Le processeur de signal est beaucoup plus spécialisé. Alors qu'un microprocesseur n'est pas conçu pour une application spécifique, le processeur DSP (Digital Signal Processor) est optimisé pour effectuer du traitement numérique du signal (calcul de FFT, convolution, filtrage numérique, etc...).

Les domaines d'application des D.S.P étaient à l'origine les télécommunications et le secteur militaire. Aujourd'hui, les applications se sont diversifiées vers le multimédia (lecteur CD, MP3, etc..), l'électronique grand public (télévision numérique, téléphone portable, etc…), l'automatique, l'instrumentation, l'électronique automobile, etc…