Ch4 – Le MicroProcesseur

Dernière maj : lundi 2 avril 2007

I. Jeu d'instruction

Le microprocesseur est le composant chargé d'exécuter les programmes situés en mémoire centrale. Il doit donc disposer d'un ensemble d'instructions afin d'être capable de réaliser ce qu'on lui demande.

A. Définition

Le **JEU D'INSTRUCTION** d'un microprocesseur est l' **ENSEMBLE DES INSTRUCTIONS CONNUES DU MICROPROCESSEUR**, chacune étant identifiée par un code binaire.

Ces instructions sont matérialisées par des **CIRCUITS SPECIALISES** localisés dans l' **U**nité **A**rithmétique et **L**ogique. Elles correspondent aux compétences du microprocesseur.

Le **FORMAT D'UNE INSTRUCTION** soumise au microprocesseur comprend 2 parties :
_une **ZONE INSTRUCTION** : le code binaire de l'instruction
_une **ZONE ADRESSE** : qui contiendra les adresses ou les valeurs des opérandes

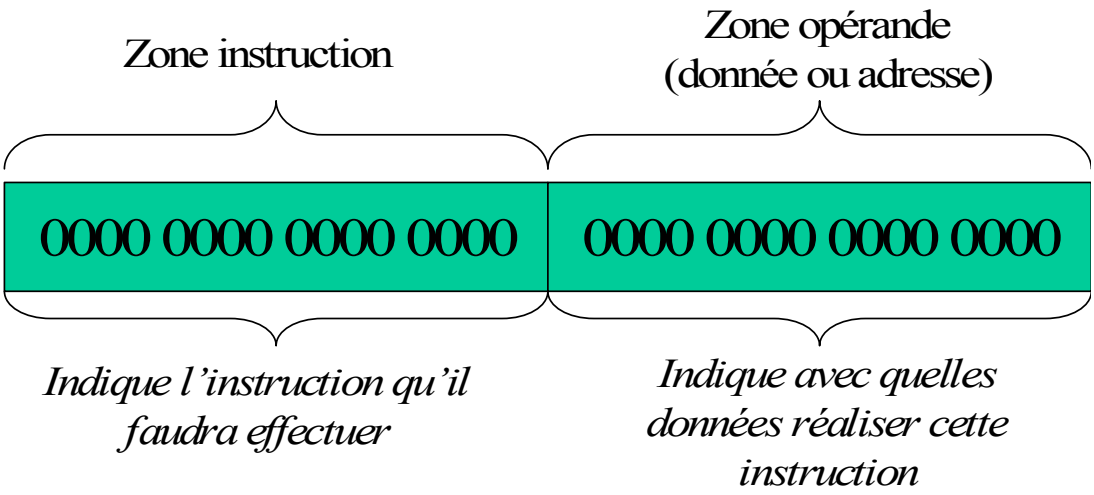

Figure 1 : champs instruction : code instruction et opérande

B. Langages et jeu d'instruction

►Le **LANGAGE MACHINE** est le langage dans lequel les **INSTRUCTIONS** des programmes sont **DIRECTEMENT EXECUTABLES**, chaque instruction étant exprimée par son code binaire. Ce langage est généralement spécifique à une famille de microprocesseurs.

►Le **LANGAGE D'ASSEMBLAGE** est un langage dans lequel les instructions du jeu d' **INSTRUCTIONS** sont exprimées par des **CODES MNEMONIQUES** plus facilement manipulables par un opérateur humain. Le programme en langage Assembleur (programme source) doit être en suite traduit en langage machine (1 instruction assembleur ➔ 1 instruction machine)

►Le **LANGAGE DE HAUT NIVEAU** (LHN), ou langage évolué, est un langage plus proche dans lequel l'opérateur humain peut concevoir des programmes indépendants du jeu d'instruction. Des compilateurs se chargeront de transformer chaque instruction LHV en plusieurs instructions assembleur.

- Le langage machine correspond donc aux instructions directement utilisables par un modèle de microprocesseur.
- Certains jeux d'instruction sont standard : par exemple les microprocesseurs Intel et AMD utilisent des jeux d'instructions sont les codes sont pour la plupart les mêmes : c'est pour cela que vos programmes tournent aussi bien sur l'un ou l'autre des microprocesseurs de ces familles.
- Par contre, d'autres constructeurs utilisent des jeux d'instructions non compatibles : vous ne pourrez exécuter un programme en langage machine Motorola sur une machine à base de microprocesseur Intel.

- Machines virtuelles (Java Virtual Machine, ou Common Language Runtime) correspondent à des nouvelles manières d'exécuter des

programmes. Au lieu de traduire tous les programmes dans le langage binaire de tel ou tel microprocesseur, on les traduira dans un langage intermédiaire (Byte Code) qui sera interprété (ou compilé) par un programme au moment de l'exécution (ce programme correspond à la machine virtuelle : c'est à lui à qui on va soumettre l'exécution d'un programme, comme à une machine).

C. Les instruction du jeu d'instruction

Les instructions utilisable par le microprocesseur ne réalisent que des fonctions basiques : chargement de valeurs de la mémoire vers les registres du microprocesseur, stockage des valeurs des registres vers la mémoire centrale, opérations arithmétiques et logiques et opérations de saut.
Un programme pourra être réalisé à partir de ces instructions de base.

Trois types d'instructions permettent la réalisation de :

►TRANSFERT MEMOIRE CENTRALE ←→ REGISTRES DU MICROPROCESSEUR
➔ Ces instructions permettent de charger des valeurs dans les registres du microprocesseur et stocker des valeurs calculées dans la mémoire.

►OPERATIONS ARITHMETIQUES ET LOGIQUES
➔ Ces instructions portent sur les additions, multiplications et autres opérations arithmétiques et sur l'application des opérateurs logiques ET, OU, etc.

►OPERATIONS DE BRANCHEMENT
➔ Ces instructions permettent de modifier la séquence linéaire du programme

- Les codes mnémoniques (langage d'assemblage) associés à ces instructions sont assez représentatifs du traitement qu'elles effectuent (en anglais):
  - LOAD, LD : chargement de la valeur d'un mot mémoire vers un registre
  - STORE, ST : stockage d'une valeur de registre vers un mot mémoire
  - ADD, MULT, SUB,… : pour additionner, multiplier, soustraire
  - OR, AND,… : pour effectuer un OU logique ou un ET logique
  - JMP, JMPZ,… : pour effectuer un branchement (un saut : anglais JUMP) à une autre adresse mémoire du programme

D. Les registres

Les registres sont au centre des échanges à l'intérieur du microprocesseur.
Ils accueillent

- des données en provenance de la mémoire en vue de leur traitement,
- des résultat des opérations arithmétiques et logiques avant en attendant quelles soient recopiées en mémoire.

►Les **REGISTRES** sont des **CELLULES MEMOIRES INTERNES AU MICROPROCESSEUR**.

Chaque registre est identifié par une adresse, qui est représentée par un nom au niveau des langages d'assemblage.

- Le nombre de registres varie selon le microprocesseur, les plus évolués en contenant un nombre important.
- Chaque registre est identifié par un numéro (code mnémonique en assembleur) et certains sont dédiés à des fonctions particulières (par exemple un registre nommé Accumulateur sert dans les opération d'additions).

II. Modes d'adressage

Les instructions et données utilisent des zones mémoires qu'on appelle segments. Plusieurs segments de programmes et de données peuvent être définis.
Les opérations utilisent des opérandes : celles-ci sont spécifiées de plusieurs manières et nécessitent donc différentes manières de retrouver l'adresse réelle en mémoire. (exemple fournis en assembleur)

►Un **MODE D'ADRESSAGE** correspond à une **METHODE DE LOCALISATION DES MOTS MEMOIRES**.

A. Les différents modes d'adressage

De la même manière que nous utilisons des formules différentes pour pouvoir exprimer un lieu où nous habitons, un code instruction pourra exprimer de différences façon l'endroit où se trouve la données qu'il va utiliser.

Modes d'adressage les plus courant :

►adressage **IMMEDIAT** : la zone adresse de l'instruction contient la donnée

►adressage SYMBOLIQUE : on affecte un nom, un symbole à une adresse et l'instruction utilisera ce nom pour adresser un mot mémoire (utilisé dans les langages d'assemblage)

►adressage **ABSOLU** / **DIRECT** : la zone adresse de l'instruction correspondra à l'adresse de la données à utiliser.

►adressage IMPLICITE / REGISTRE : la zone adresse de l'instruction contient un numéro de registre

►adressage **RELATIF** / INDEXE : la zone adresse de l'instruction contient un nombre correspondant à un déplacement (offset) par rapport à l'adresse actuelle (du compteur ordinal)

►adressage **INDIRECT** : la zone adresse de l'instruction contient une valeur qui représente une adresse à laquelle on pourra trouver l'adresse de la données…

- Un programme correspond à une tâche à effectuer ;
- Chaque instruction d'un programme correspond aux différentes phases élémentaires d'une tâche
- L'unité d'échange correspond à la main tendue qui prend du demandeur la feuille et qui la lui rend
- Le bus va correspondre au système nerveux ainsi que les sens associés : la vue permet de communiquer les instructions de la feuille au cerveau, le système nerveux permet de commander l'action d'écrire le résultat sur la feuille

III.Les Interruptions

Combien de fois sommes-nous interrompus alors que nous réalisons une tâche : parfois il est important se suspendre ce que nous étions en train de faire et prendre en compte la nouvelle demande, plus urgente par exemple.

Le microprocesseur doit prend en compte ce mécanisme : en effet, essayons d'imaginer aujourd'hui un ordinateur qui effectuerait une tâche d'impression de 50 pages non interruptible, avant de pouvoir enchaîner sur une navigation sur Internet…

A. Définition.

►L'**INTERRUPTION** correspond au fait de signaler au microprocesseur qu'il doit interrompre l'exécution du programme en cours afin d'exécuter le programme demandé.

►La **DEMANDE D'INTERRUPTION** (anglais Interrupt ReQuest, **IRQ**) est identifiée par un numéro ; plusieurs numéros sont disponibles, chacun étant affecté à un type d'interruption déterminé. Chaque interruption est traitée par un programme situé en mémoire centrale (vecteur d'interruptions).

B. Les types d'interruptions

Plusieurs types d'interruptions :

►Interruptions **MATERIELLES externes** : à la demande d'un périphérique (clavier, souris,…)

►Interruptions **LOGICIELLES externes** : à la demande d'un programme afin d'effectuer un appel à des routines systèmes (pilotes, BIOS,…)

►Interruptions **INTERNES** ou **EXCEPTIONS** (anglais : TRAP) : liées au fonctionnement du microprocesseur, suite à la détection d'une anomalie dans l'exécution d'une instruction (division par zéro) ou d'un problème de fonctionnement

- Une routine est un programme qui fait partie du système d'exploitation (ou des autres logiciels systèmes) et qui réalise une tâche particulière : il existe, par exemple, des routines d'accès aux disques durs.

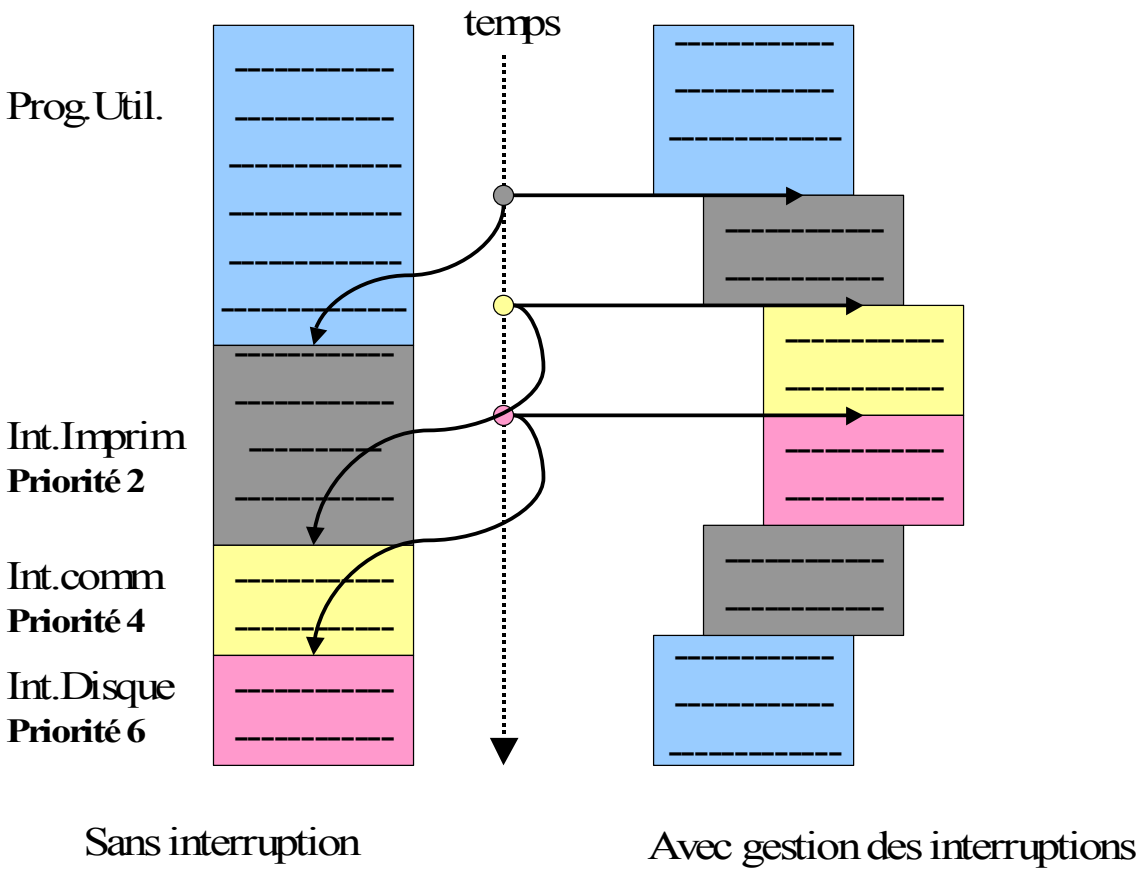

Figure 2 : les interruptions permettent de fractionner l'exécution de tâches en fonction de l' urgence, la priorité d'autres tâches à réaliser

C. Hiérarchisation des interruptions

Au même titre que nous évaluons le degré d'urgence des interruptions qui nous parviennent afin de n'interrompre une tâche qui si l'exécution d'une tâche plus urgente est demandée, le microprocesseur associe des niveaux de priorités aux interruptions selon l'importance relatives qu'elles ont les unes par rapport aux autres.

De plus, l'interruption sera définie comme **non masquable** (on ne peut ignorer sa présence) ou bien **masquables** (on peut l'ignorer pour un moment).

Hiérarchisation des interruptions :

►les interruptions **NON MASQUABLES** seront traitées **IMMEDIATEMENT**, dès que l'instruction en cours est terminée (exemple : problème matériels)
►les interruptions **MASQUABLES** seront traitées **PLUS TARD**, après la fin d'un programme (exemple : fin de papier)

La **PRIORITE** va préciser pour chaque type d'interruption, si elle est prioritaire par rapport à une autre, si son exécution est plus urgente qu'une autre interruption.

D. L'exécution des interruptions

De même que nous traitons une interruption en sauvegardant le travail en cours pour le reprendre plus tard, le microprocesseur devra sauvegarder l'état actuel d'un programme afin de le reprendre après l'exécution du programme lié à une interruption.
On peut signaler également que le microprocesseur terminera l'exécution de l'instruction en cours avant de prendre en charge une interruption.

Afin d'éviter de trop encombrer le microprocesseur en ayant autant de lignes d'interruptions qu'il y a de possibilités d'interruptions (donc de périphériques pouvant interrompre le microprocesseur), un contrôleur d'interruption prend en charge la sélection de l'interruption pour n'envoyer au microprocesseur qu'une seule ligne d'interruption.

Phases dans l'exécution d'une interruption

1-**RECEVOIR** le **TYPE D'INTERRUPTION** (et éventuellement l'adresse du périphérique)
2-**SAUVEGARDER** l'état du **PROGRAMME EN COURS**
3-**APPELER** la **ROUTINE** capable de gérer l' **INTERRUPTION**
4-**EXECUTER** cette **ROUTINE**
5-**RESTAURER** l'état du **PROGRAMME**
6-**POURSUIVRE** l'exécution du **PROGRAMME**

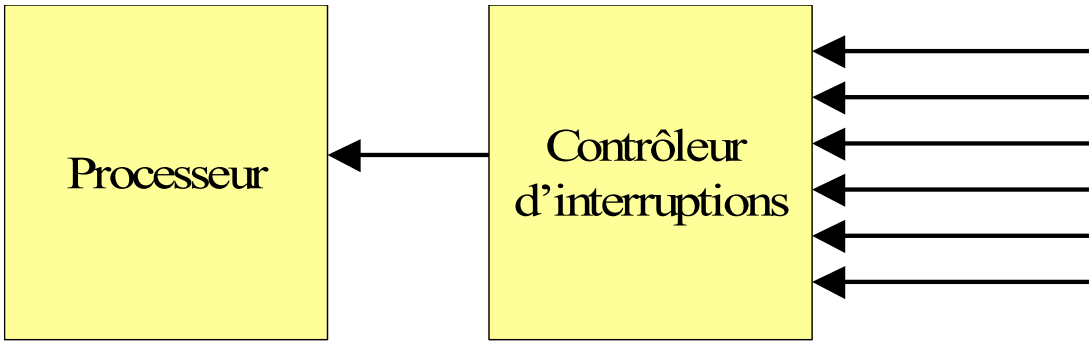

Figure 3 : le contrôleur prend en charge le flux des interruptions

IV. Performances et Améliorations des performances

A. Caractéristiques et performances d'un microprocesseur

Comme nous l'avons vu précédemment, le cadencement de l'exécution des instructions est dirigé par une horloge : c'est un des indicateurs de performance affichés.

Caractéristiques :

►**FREQUENCE** d'exécution des instructions exprimée en multiples de Hz
➔ Actuellement env. 2 à 3 GHz (1GHz = un milliard de fois par seconde)

►**NOMBRE ET TAILLES DES REGISTRES** : un nombre de registres élevés permettra de limiter l'accès à la mémoire
➔ Actuellement env. 100 à 200 registres

►**TAILLE DU BUS INTERNE** : une largeur de bus interne élevée permettra le transport de plus de données en même temps par cycle d'horloge
➔ Actuellement env. 64 à 128 bits

►**MEMOIRES CACHES** : zones de mémoire internes évitant des accès mémoire au microprocesseur
➔ Actuellement env. 1Mo à 2Mo

- Cependant, pendant un certain laps de temps (un cycle, soit un battement d'horloge), tous les microprocesseurs n'exécutent pas le même nombre de micro-instructions : la fréquence de l'horloge est donc à relativiser.

Indicateurs de performances :

►**NOMBRE D'INSTRUCTIONS PAR CYCLE D'HORLOGE** : permet d'établir des échelles de grandeurs pour **COMPARER DES PERFORMANCES DE MICROPROCESSEURS** :
➔ MIPS (millions d'instructions par secondes)
➔ MFLOPS (millions d'instructions flottantes par secondes)
➔ IPC (nombre d'Instructions Par Cycle), etc.

►**BENCHMARK** (français : banc d'essai) : permet d'établir des échelles de grandeurs pour **COMPARER DES PERFORMANCES DE CONFIGURATIONS DE MICROORDINATEURS** selon des profils particuliers :
➔ Orientés bureautique
➔ Orientés graphiques ou jeu
➔Orientés serveurs de données, etc.

- **La performance d'un microordinateur n'est pas seulement liée au seul microprocesseur ; c'est donc à travers des configurations d'architecture complètes qu'il sera possible d'effectuer des comparatifs.**

B. Amélioration des performances : fréquence

Améliorations des performances internes:

►Augmenter les valeurs des caractéristiques du microprocesseur (c'est ce que font les constructeurs de microprocesseurs)

►**OVERCLOCKING** : augmentation de la fréquence du microprocesseur

- L'overclocking est une opération délicate qui nécessite parfois une augmentation des tensions électriques afin de les signaux 0 ou 1 puissent être correctement détectés. Cette augmentation provoque une **surchauffe qui peut endommager physiquement le microprocesseur** si sa qualité n'est pas excellente.

C. Amélioration des performances : technologies

Différentes orientations technologiques ont été mises en œuvre dans la conception des microprocesseurs. Les constructeurs ont essayés de se différencier se concentrant vers l'une ou l'autre de ces technologies.

Technologies :

►**CISC (Complex Instruction Set Computing) :**

➔ Jeu d'instruction important : 200-300 instructions
➔ Plusieurs modes d'adressage : env. 10
➔ Format des instructions variable (temps de décodage long)
➔ Nombre de registres peu important : env. 50 registres
➔ Exemple INTEL, AMD, MOTOROLA

►**RISC (Reduced Instruction Set Computing) :**
➔ Jeu d'instruction réduit : env. 100 instructions
➔ Un seul mode d'adressage
➔ Format des instructions fixe (temps de décodage court)
➔ Nombre de registres important : env. 100 registres
➔ Exemple SUN SPARC

- Ces 2 orientations apportent leurs avantages et inconvénients, si bien qu'aujourd'hui les technologies dites CISC intègrent un peu de RISC dans leur fonctionnement, et vice versa.

D. Amélioration des performances : architectures

L'augmentation de la fréquence des microprocesseurs atteint aujourd'hui des sommets qui commencent à apporter leurs lots de difficultés : l'augmentation de la température nécessite des systèmes de refroidissement plus complexe, la consommation électrique des microprocesseurs augmente, etc.

Des ingénieurs ont mis en œuvre des solutions qui en leur temps ont permis l'augmentation de la productivité des activités humaines : la DIVISION DU TRAVAIL et le travail à la chaîne. Cela correspond au fait de découper le travail en plusieurs parties, chacune étant réalisé en même temps.

Architectures :

►**PIPELINE** permet l'exécution de **PLUSIEURS INSTRUCTIONS A LA FOIS**
➔ Découpage d'une instruction en micro-instructions, chacune étant exécutée dans une partie du microprocesseur
➔ Nombre d'étages déterminé par le découpage (nombre d'instructions exécutées simultanément)

►**SUPERSCALAIRE** permet l'exécution de **PLUSIEURS OPERATIONS** simultanément **DANS L'UAL**

►**MULTICORE** : un microprocesseur contient plusieurs unités d'exécution complète (DUAL-CORE, QUAD-CORE)

- La technologie **PIPELINE** a cependant des limites : au fur et à mesure que le nombre d'étages augmente, il est nécessaire de mettre en œuvre des mécanismes de prédiction pour que les instructions puissent s'exécuter en même temps sans risque (problèmes des branchements conditionnels, opérations sur des variables utilisées dans plusieurs instructions, etc.) : **la performance ne croit pas aussi vite que prévu**.
- La technologie MULTICORE n'est pas exploitée pleinement actuellement car elle nécessite une prise en compte par les logiciels

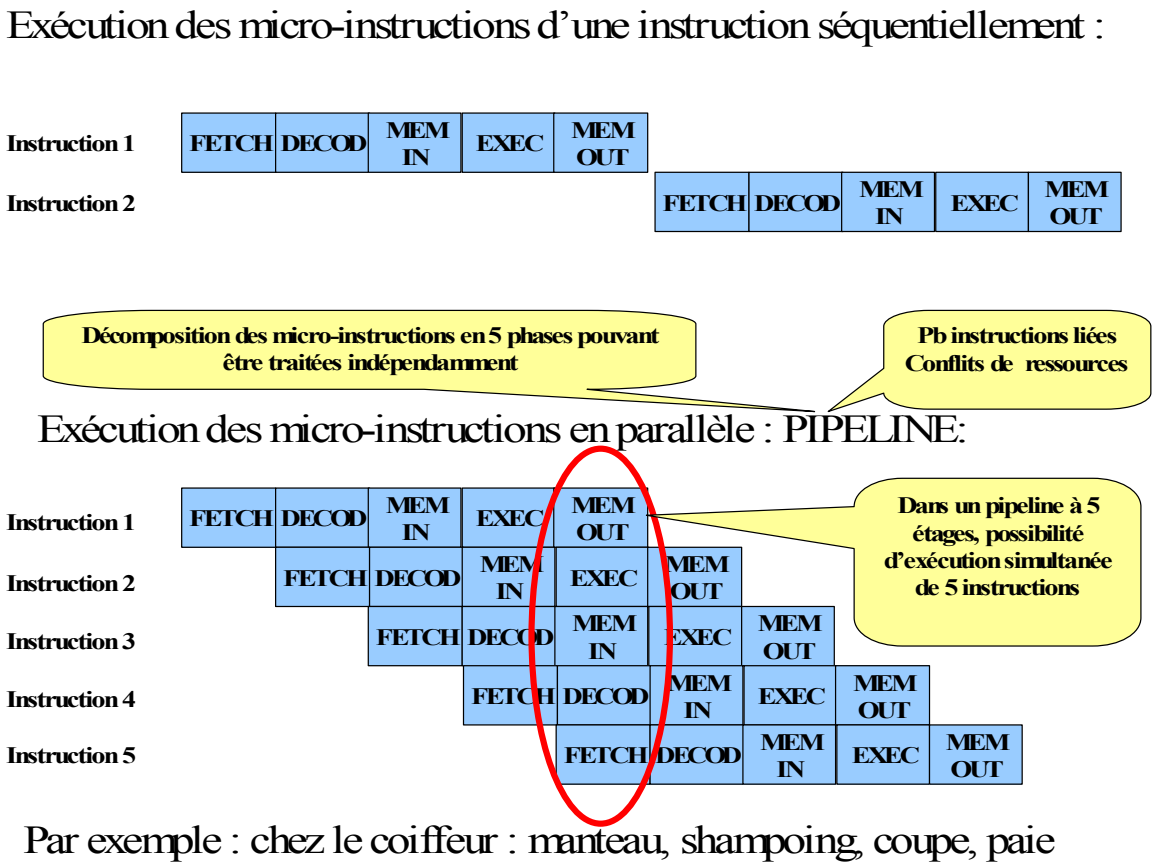

Figure 4 : principe du PIPELINE : plusieurs instructions exécutées simultanément

E. Amélioration des performances : multiprocesseurs

Technologies :

►**MULTIPROCESSEURS** : plusieurs processeurs indépendants
➔ Le système d'exploitation doit être capable de prendre en compte de type de configuration
➔ Actuellement, par exemple les cartes mères à intégrant un chipset capable de gérer 2, 4 ou 8 microprocesseurs Intel Xéon (réservé aux serveurs)

►**RESEAUX D'ORDINATEURS** : plusieurs ordinateurs indépendants forment un supercalculateur ; un réseau ultra-rapide connecte ces ordinateurs

➔ Ces calculateurs forment ce qu'on appelle des grilles de calcul (**GRID COMPUTING**) et permettent de résoudre des problèmes complexes (recherche génétique, etc.)

- Un tableau des familles d'ordinateurs a été ainsi établi :
  - Monoprocesseurs :
    - SISD : Single Instruction Single Data
    - SIMD: Single Instruction Multiple Data
  - Multiprocesseurs :
    - MISD : Multiple Instruction Single Data (partage de la mémoire entre les processeurs)
    - MIMD : Multiple Instruction Multiple Data (chaque processeur gère sa mémoire).

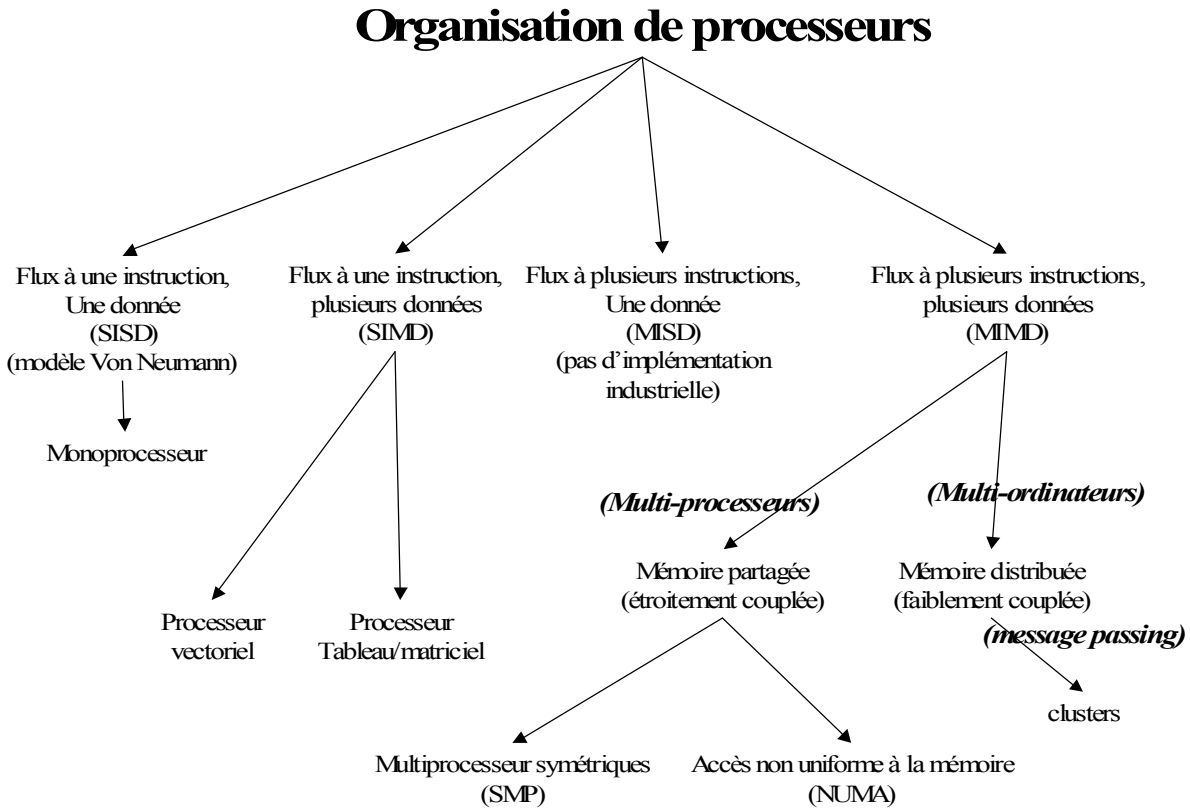

Figure 5 : taxinomie des machines à base de processeur

Par exemple pour la ligne Intel Centrino

Voici quelques-unes des fonctionnalités qui contribuent à ses performances exceptionnelles :

* La fusion des micro-opérations, c'est-à-dire l'association de certaines opérations pour réduire le délai et l'énergie nécessaire à l'exécution des instructions.
* Une mémoire cache de niveau 2, à optimisation électrique, et une unité améliorée de pré-extraction des données, qui réduisent les accès mémoire en dehors de la puce et qui assurent la présence en mémoire cache de niveau 2 d'une plus grande proportion de données valides.
* Une prédiction de branchement ultraperfectionnée, qui, en analysant les comportements antérieurs et en prévoyant les opérations susceptibles d'être sollicitées ultérieurement, élimine les traitements en double pour le processeur.
* baisse de la tension pendant le temps où le processeur est moins utilisé.