



DIRO  
IFT 1215

# INTRODUCTION AUX SYSTÈMES INFORMATIQUES

## LE CPU ET LA MÉMOIRE II

*Max Mignotte*

Département d'Informatique et de Recherche Opérationnelle  
Http: [//www.iro.umontreal.ca/~mignotte/](http://www.iro.umontreal.ca/~mignotte/)  
E-mail: [mignotte@iro.umontreal.ca](mailto:mignotte@iro.umontreal.ca)

## LE CPU ET LA MÉMOIRE II

### SOMMAIRE

Introduction .....	2
Architecture CISC-RISC .....	2
Architecture VLIW-EPIC .....	5
Mémoire .....	7
Paging .....	10
Amélioration reposant sur l'accès mémoire ...	14
Entrelacement de Mémoire .....	15
Mémoire Cache .....	16
Méthodes de Traitements Modernes .....	21
Pipeline .....	22
Processeur scalaire/SuperScalaire .....	24
Processeur SuperScalaire .....	25
Séquenceur Micro-programmé .....	26

# LE CPU ET LA MÉMOIRE II

## INTRODUCTION - ARCHITECTURE CISC-RISC

### Introduction

On discutera dans ce chapitre des techniques et caractéristiques modernes qui permettent de donner aux ordinateurs actuels toute leur puissance

### Architecture CISC & RISC

- CISC - [*Complex Instruction Set Computer*]
- RISC - [*Reduced Instruction Set Computer*]

Chacune de ces deux architectures est consistante avec les caractéristique d'une architecture selon Von Neumann

### CISC

**Exemple :** Famille Intel x86 et AMD, Ordinateur IBM serie Z, la plupart des plus vieille architectures existantes, etc.

### Caractéristiques :

1. Un grand nombre d'instructions spécialisées et complexes
2. Instructions de tailles différentes
3. Peu de registre [*general purpose register*] dans le CPU
4. Beaucoup de mode d'adressages différents

### Limitations :

- Les instructions complexes ne sont que rarement utilisés
- Le faible nombre de registre nécessite de faire appelle à la mémoire qui ralentit considérablement le CPU

# LE CPU ET LA MÉMOIRE II

## ARCHITECTURE CISC-RISC

### RISC

Le concept RISC est de la constatation suivante:

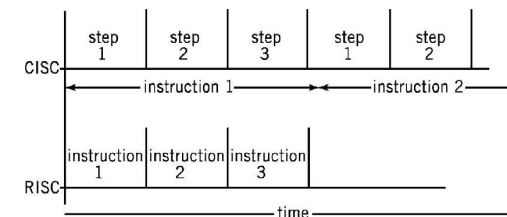
Dans 80 % des cas, un processeur n'utilise que 20 % de son jeu d'instructions

	Instruction	% of Executions
BC	Branch condition	20.2
L	Load	15.5
TM	Test under mask	6.1
ST	Store	5.9
LR	Load register	4.7
LA	Load address	4.0
LTR	Test register	3.8
BCR	Branch register	2.9
VC	Move characters	2.1
LH	Load halfword	1.8

**Exemple :** Power PC, Sun Sparc, Famille Motorola, etc.

### Caractéristiques :

1. Jeu restreint d'instructions simples effectuées rapidement en un cycle d'horloge
2. Instruction de taille fixe
3. Beaucoup de registres (utilisation intensive réduisant l'accès à la mémoire)
4. Peu de mode d'adressages différents



## LE CPU ET LA MÉMOIRE II

### ARCHITECTURE CISC-RISC

#### RISC vs CISC

**RISC** : Moins d'instructions et instructions plus simples

- Simplifie le matériel pour augmenter les performances en utilisant une nouvelle génération de compilateur

**CISC** : Plus d'instruction mais instructions plus complexes

- Simplifie la tâche du compilateur (logiciel)  
- Plus de registre améliorerait les performance de l'architecture CISC mais pas assez de place sur la puce !

La guerre entre CISC et RISC n'a pas réellement eu lieu

Les approches CISC et RISC migrent vers les mêmes objectifs

L'exécution d'une ou plusieurs instructions en un cycle d'horloge avec des fréquences de plus en plus rapide (les deux utilisent les même principe de memoire cache, pipeline, gestionnaire de mémoire, etc.) (Ex: Pentium)

## LE CPU ET LA MÉMOIRE II

### ARCHITECTURE VLIW - EPIC

#### Architecture VLIW - EPIC

- VLIW - [*Very-Long Instruction Word*]
- EPIC - [*Explicitely Parallel Instruction Computer*]

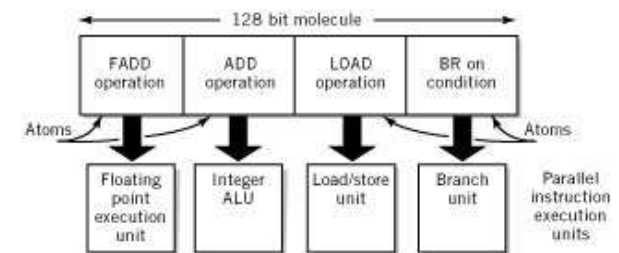
Le but de ces deux type d'architecture est d'augmenter la vitesse d'exécution du processeur en traitant des instructions/opérations en parallèle

#### VLIW

**Exemple** : CPU Transmeta Crusoe

**Caractéristiques** :

1. Instruction de longueur 128 bits : *molécule* divisé en 4 *atomes* de 32 bits fournissant 4 opérations pouvant être exécuter simultanément



2. 64 registres
3. Logiciel de Codage Morphing  
Traduction du code machine d'autre CPU en *molécules* (intégré à l'architecture)

## LE CPU ET LA MÉMOIRE II

### ARCHITECTURE VLIW - EPIC

#### EPIC

**Exemple :** CPU Intel Itanium

Même but mais avec des caractéristiques différentes

**Caractéristiques :**

1. Instruction de longueur 128 bits comprenant 3 instructions de 41 bits + 5 bits pour identifier le type d'instruction
2. Les instructions sont les mêmes que ceux existant pour la même famille de processeur (x86) non EPIC existant
3. Les 5 bits sont des bits d'informations qui permettent d'identifier les dépendances potentielles entre exécutions

#### EPIC vs VLIW

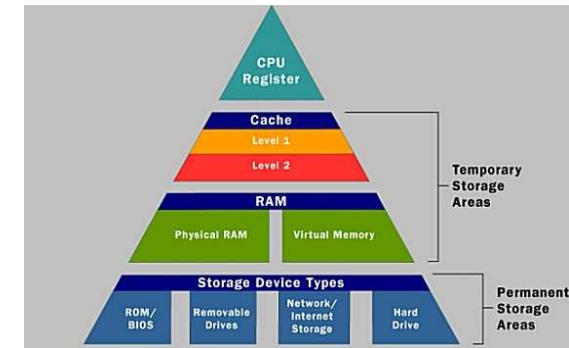
Le séquençement des opérations (+ gestion des priorités, dépendance, etc. ) dans l'instruction de 128 est

1. Intégré dans l'architecture matériel pour le VLIW
2. Géré par le programmeur ou le compilateur (logicielle)

## LE CPU ET LA MÉMOIRE II

### MÉMOIRE

#### Hiérarchie de Mémoire

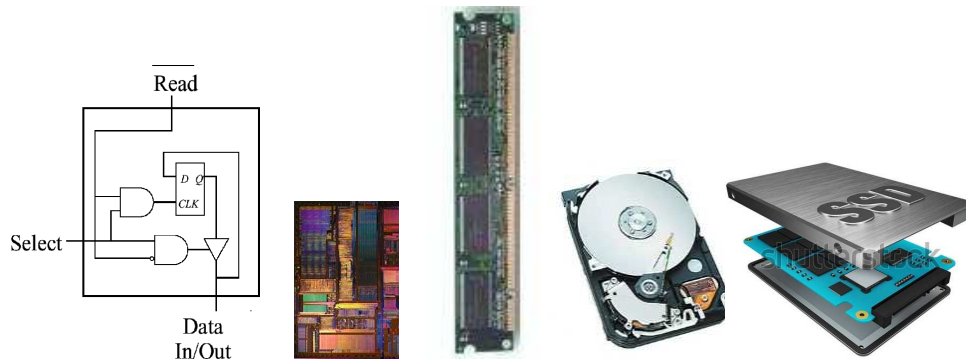


Memory Type	Access Time	Typical Amount Used
Registers	0.25 - 0.5 ns	1 KB
Cache	1 - 5 ns	1 - 10 MB
Main Memory	10 ns - 50 ns SRAM    DRAM	10 GB
Disk Memory	5 ms - 25 $\mu$ s hard disk    SSD	1 TB

- **Registres** interne au CPU, rapide et serve au stockage des instructions et résultats intermédiaires
- **Mémoire Cache** (antémémoire) interne au CPU, mémoire tampon permettant au CPU de faire moins d'accès mémoire
- **Mémoire centrale** organe principal de rangement des données par le CPU
- **Mémoire de Masse** utilisé pour le stockage, sauvegarde, archivage (support magnétique, magneto-optique ou optique)

## LE CPU ET LA MÉMOIRE II

### MÉMOIRE



• RAM [*Random Access Memory*] tps d'accès indépendant du numéro de la cellule

1. **SRAM** [*Static Ram*] Chaque point mémoire nécessite quatre transistors, très rapide (registre)
2. **DRAM** [*Dynamic Ram*] les données doivent être rafraîchit. Un point mémoire est constitué d'un transistor couplé à un condensateur

Fabrication plus simple, densité d'intégration plus grande, par contre plus lent

• ROM [*Read Only Memory*] on peut lire uniquement, écriture possible (stockage de certain programme système)

• SSD [*Solid State Drive*] est un dispositif de stockage mémoire qui utilise des assemblages de circuits intégrés pour stocker des données de manière persistante, généralement à l'aide d'une mémoire flash.

## LE CPU ET LA MÉMOIRE II

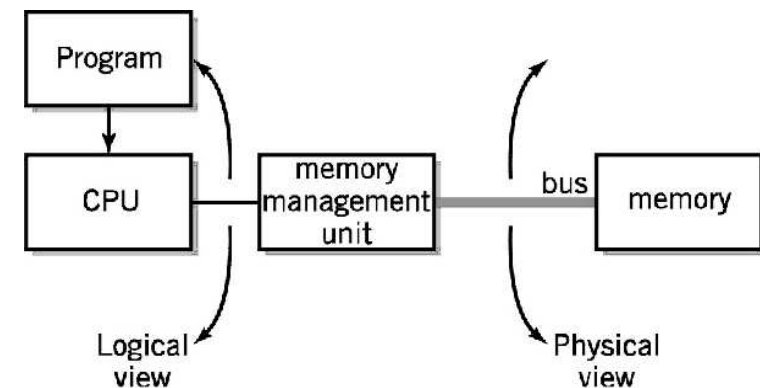
### PAGING

#### Paging

Technique qui permet la réalisation du concept de mémoire virtuelle

Consiste à séparer conceptuellement (& traiter séparément) les adresses utilisées dans un programme (adresses virtuelles) et les adresses de la mémoire physique (adresses réelles)

1. Géré par l'OS
2. Construit matériellement (hardware)
3. Indépendant de l'application

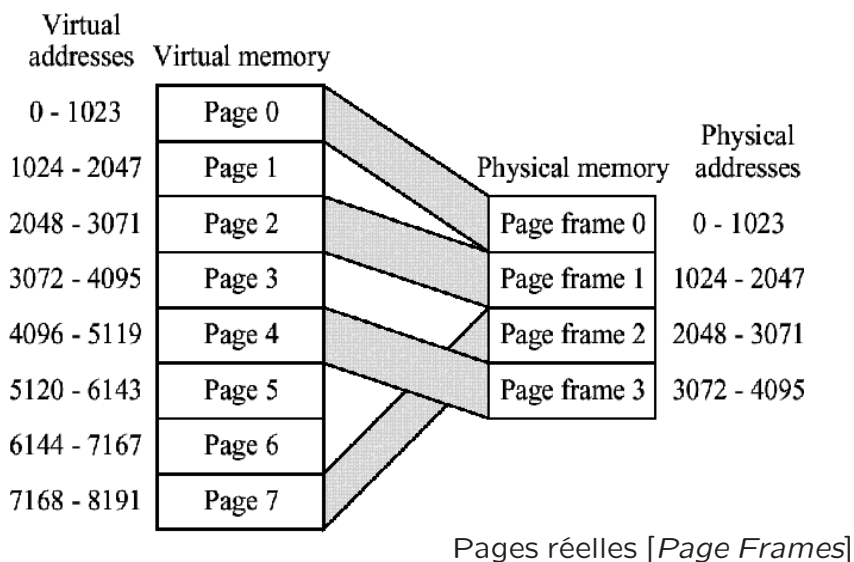


Dans ce modèle, tout se passe comme si on avait à disposition une mémoire plus grande que la taille de la mémoire physique

## LE CPU ET LA MÉMOIRE II PAGING

La Pagination [*Paging*] rend la mémoire physique plus grande qu'elle n'est réellement sans que le programmeur n'ai à s'en soucier

Ex: Ordi. 13 bits (> 8 KOctets adressable) et 4 KOctets de mémoire réelle mais suffisamment de mémoire secondaire



Le concept de pagination consiste à découper les deux espaces adresses en page de même taille et à mettre en oeuvre un mécanisme de transfert de page

## LE CPU ET LA MÉMOIRE II PAGING

### Table des Pages

La table des pages [*Page table*] (géré par l'OS) fait correspondre à chaque page virtuelle toute une série d'information régulièrement mise à jour

Page #	Present bit	Disk address	Page frame
0	1	01001011100	00
1	0	11101110010	xx
2	1	10110010111	01
3	0	00001001111	xx
4	1	01011100101	11
5	0	10100111001	xx
6	0	00110101100	xx
7	1	01010001011	10

Present bit:  
0: Page is not in physical memory  
1: Page is in physical memory

- Un bit indicateur signale si la page à été modifié pendant l'exécution et doit être recopié en mémoire aux.

### Pagination à la demande

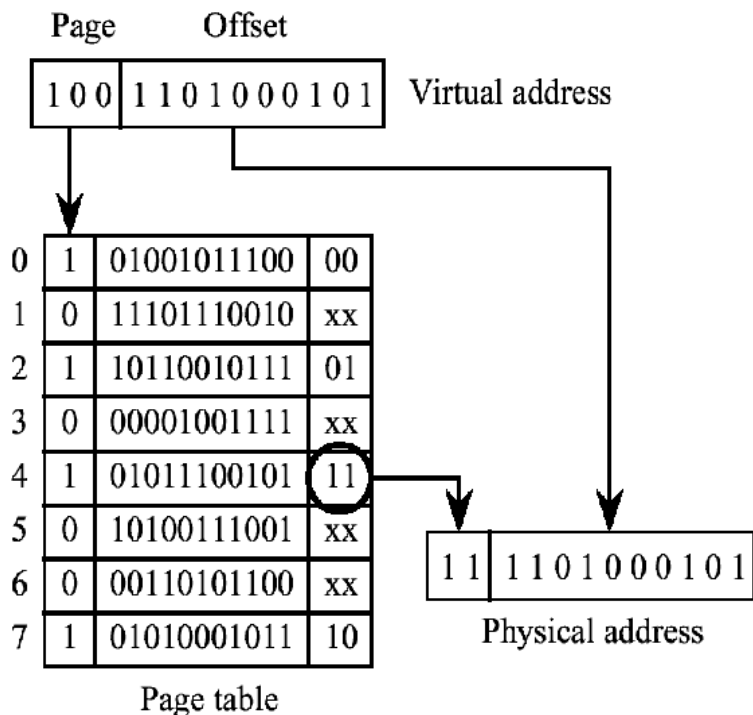
Lorsqu'une adresse référencée appartient à une page n'ayant pas de copie en mémoire > défaut de page ([*Page default*])



l'OS cherche sur disque une copie de la page demandée, la charge en mémoire

## LE CPU ET LA MÉMOIRE II PAGING

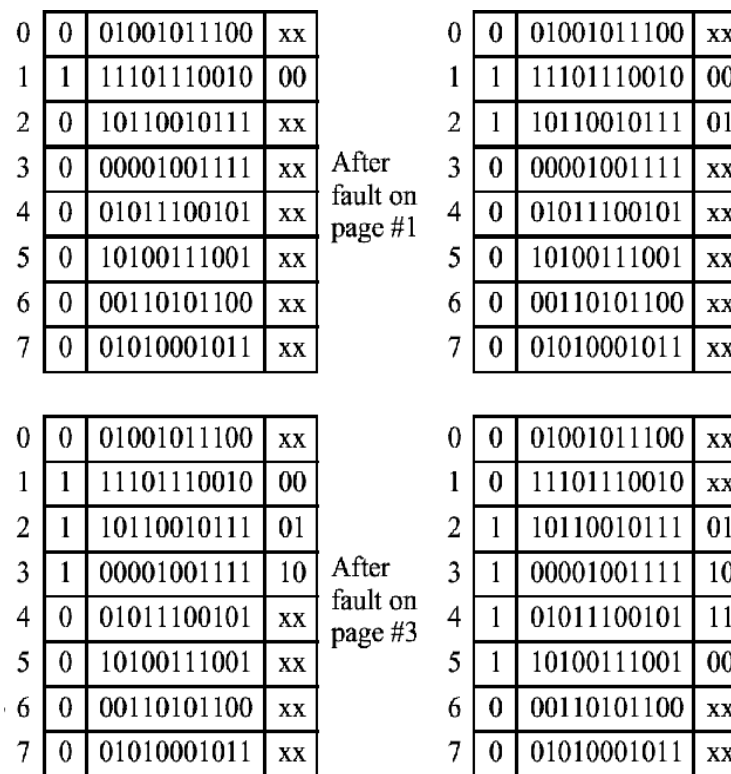
Dans le cas où une adresse référencée appartient  
à une page ayant une copie en mémoire



Une adresse virtuelle est convertit en adresse physique

## LE CPU ET LA MÉMOIRE II PAGING

Configuration d'une table de pages durant  
l'exécution d'un programme



- Initialement, la table des pages est vide
- Dans la config. finale de cet exemple, 4 pages sont chargées en mémoire physique

## LE CPU ET LA MÉMOIRE II

### AMÉLIORATION REPOSANT SUR L'ACCÈS MÉMOIRES

#### Amélioration reposant sur l'accès Mémoire

• Un accès mémoire est lent comparativement à la vitesse de processeur

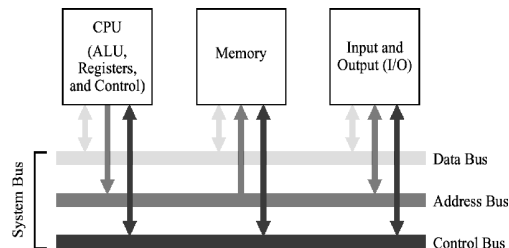
1. CPU 2 Ghz    ▷ 1 cycle en 0.5 ns
2. 20 ns DRAM    ▷ 1 accès en 40 cycle
3. 10 ns SRAM    ▷ 1 accès en 20 cycle

L'approche RISC diminue les accès mémoire par l'utilisation intensive de registre

Méthode pour diminuer le temps d'accès à la mémoire

- Accès à la mémoire avec un bus de donnée plus grand [*Wide Path Memory Access*]
- Entrelacement de Mémoire [*Memory Interleaving*]
- Mémoire Cache [*Cache Memory*]

#### Wide Path Memory Access

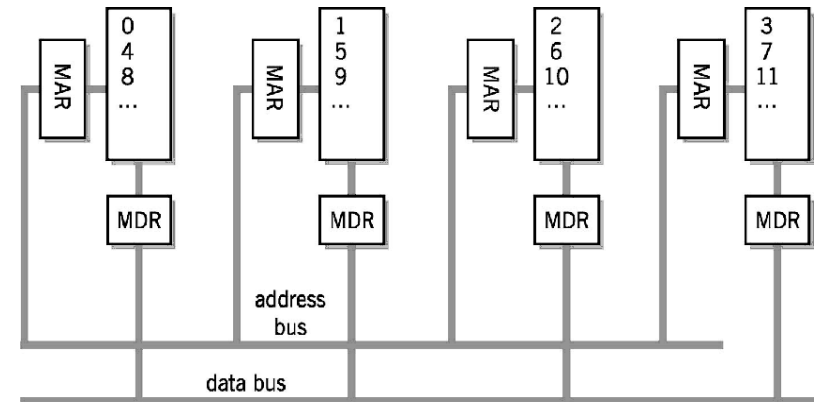


Bus de donnée plus large (4,8, Bytes etc.)

## LE CPU ET LA MÉMOIRE II

### ENTRELACEMENT DE MÉMOIRE

#### Entrelacement de Mémoire



Consiste à diviser la mémoire en  $n$  partie [ $n$ -way interleaving]

$n$  Octets [*Bytes*] peuvent être transmis en

- $\approx [ n + (\text{Tps d'accès mémoire}) ]$  cycles d'horloge au lieu de
- $\approx [ n * (\text{Tps d'accès mémoire}) ]$  cycles d'horloge



## LE CPU ET LA MÉMOIRE II

### MÉMOIRE CACHE

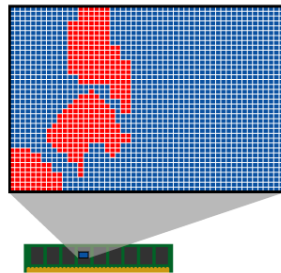
#### Introduction

**Rq. 1 :** Même le plus rapide des disques dur a un temps d'accès de 5 millisecondes

Avec un CPU de 2 Ghz, le CPU attendant 5 ms **gaspille 10 millions de cycles d'horloges !**

**Rq. 2 :** Typiquement, 90 % du temps d'exécution d'un programme est dépensé dans juste 10 % du code **principe de localité**

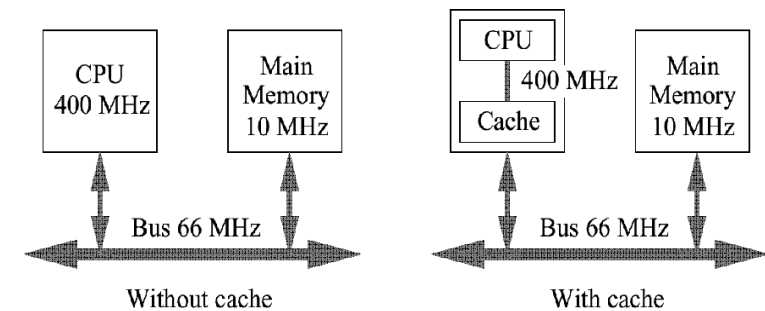
- Localité Temporelle  
*Une cellule mémoire référencé a plus de chance d'être référencé encore une autre fois (itération, récursion)*
- Localité Spatiale  
*Une cellule mémoire voisine a plus de chance d'être référencé (donnés stockés continûment)*



## LE CPU ET LA MÉMOIRE II

### MÉMOIRE CACHE

#### Mémoire Cache

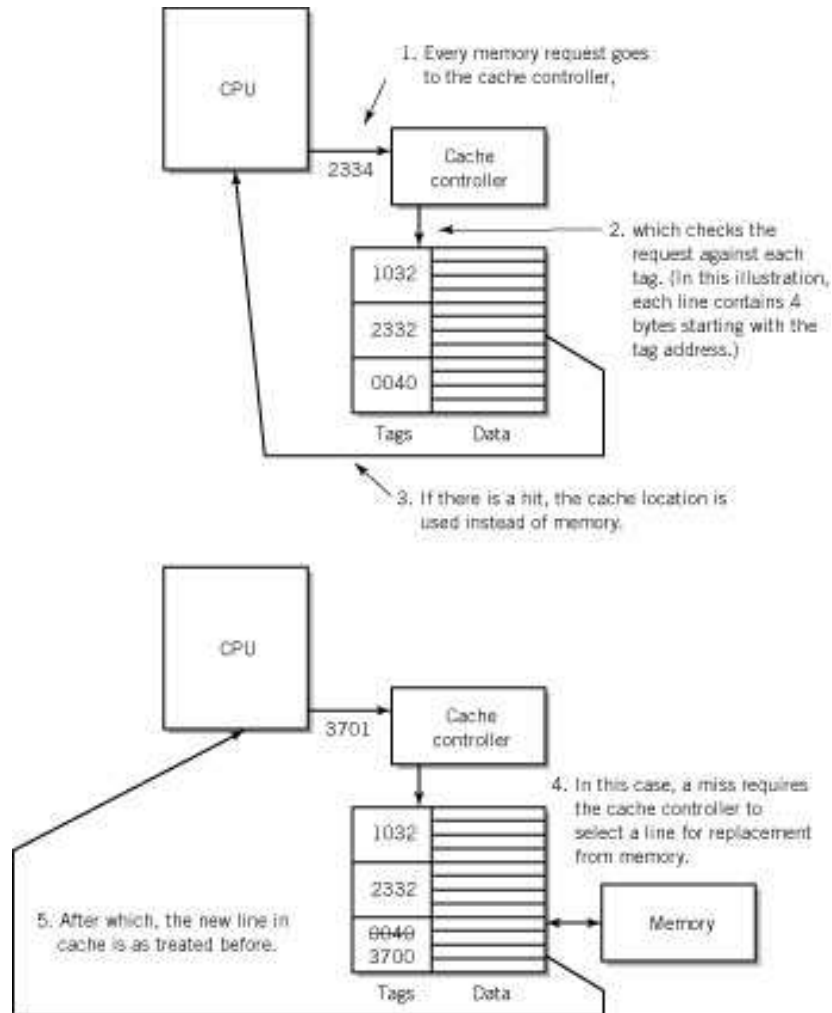


**Principe :** Ajout d'un bloc mémoire rapide dans le CPU

- Divisé en Blocs de 8 ou 16 Bytes
- Un contrôleur hardware vérifie une marque (*[tag]*) si la mémoire demandée est stockée dans la cache (*[hit]*)  
Dans le cas contraire, défaut de bloc dans le cache *[cach miss]*
- *[Cache line]* Unité de transfert entre la mémoire et la cache
- *[Hit ratio]* rapport des (*[hit]*) sur le nombre total de requêtes
- Remplacement d'un bloc lorsque la cache est pleine et que l'on a un *[miss]*
- Synchronisation Cache & Mémoire  
*[Write through] & [Write back]*

## LE CPU ET LA MÉMOIRE II MÉMOIRE CACHE

### Utilisation de la Mémoire Cache



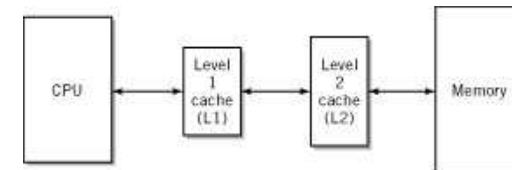
18

## LE CPU ET LA MÉMOIRE II MÉMOIRE CACHE

### Performance de la Mémoire Cache

- [Hit ratio] de 90 % couramment obtenu
- Gain de plus de 50 % en rapidité d'exécution
- Technique de mémoire cache (1 niveau) utilisé dans les disques durs

### Plusieurs niveaux de cache



- Pour être utile, le second niveau de cache doit avoir plus de mémoire que le premier niveau

### Echange entre mémoire

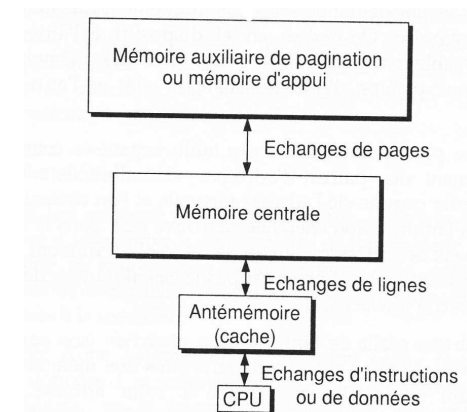


Figure 12.10 : Echanges entre les mémoires

19

## LE CPU ET LA MÉMOIRE II MÉMOIRE CACHE

### Exemple de Calcul de Hit ratio et tps d'accès effective

$$\text{Hit ratio} = \frac{\text{No. times referenced words are in cache}}{\text{Total number of memory accesses}}$$

$$\text{Eff. access time} = \frac{(\# \text{ hits})(\text{Time per hit}) + (\# \text{ misses})(\text{Time per miss})}{\text{Total number of memory access}}$$

Event	Location	Time	Comment
1 miss	48	25 ns	Memory block 3 to cache slot 3
15 hits	49-63	.8 ns×15=12 ns	
1 miss	64	25 ns	Memory block 4 to cache slot 0
15 hits	65-79	.8 ns×15=12 ns	
1 miss	80	25 ns	Memory block 5 to cache slot 1
15 hits	81-95	.8 ns×15=12 ns	
1 miss	15	25 ns	Memory block 0 to cache slot 0
1 miss	16	25 ns	Memory block 1 to cache slot 1
15 hits	17-31	.8 ns×15=12 ns	
9 hits	15	.8 ns×9=7.2 ns	Last nine iterations of loop
144 hits	16-31	.8 ns×144=115.2 ns	Last nine iterations of loop
Total hits = 213    Total misses = 5			

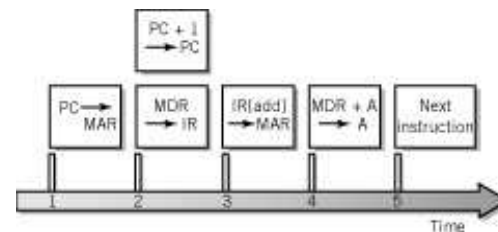
$$\text{Hit ratio} = \frac{213}{213 + 5} = \frac{213}{218} \approx 97.7\%$$

$$\text{Effective Access Time} = \frac{(213)(0.8ns) + (5)(25ns)}{213 + 5} \approx 1.36ns$$

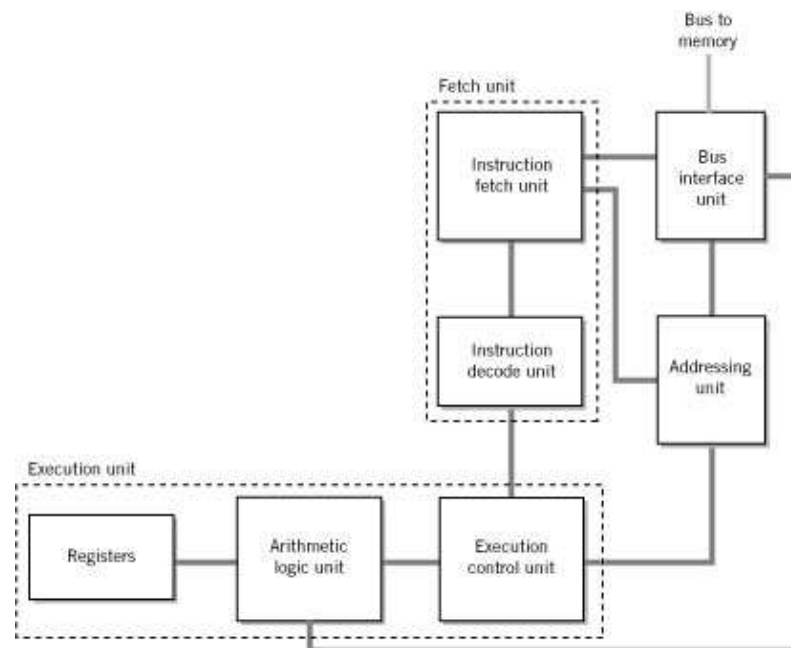
## LE CPU ET LA MÉMOIRE II MÉTHODES DE TRAITEMENTS MODERNES

### Unités *FETCH* et *EXECUTE* séparées

Cycle Fetch/Execute pour l'instruction ADD du LMC



Cycle Fetch/Execute séparées



## LE CPU ET LA MÉMOIRE II

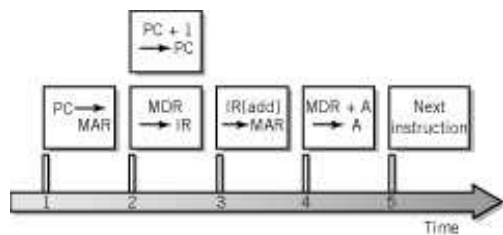
### PIPELINE

#### Pipeline

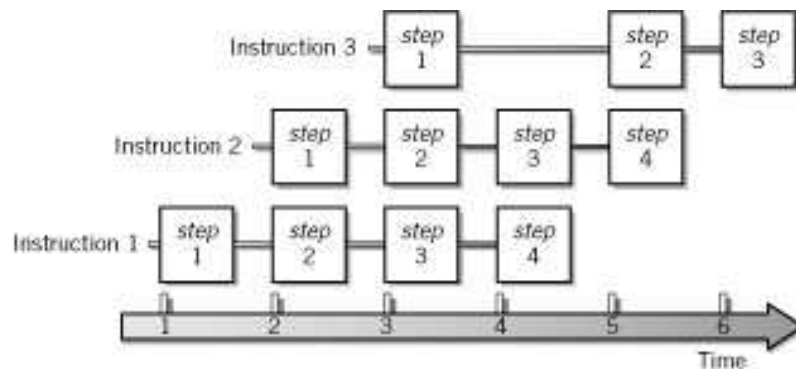
Idée inspirée de l'organisation du travail à la chaîne (e.g., chaîne d'assemblage des voitures)

- La séquence d' instructions fetch/execute est exécutée comme si celle-ci était dans une chaîne de montage

Cycle Fetch/Execute pour l'instruction ADD du LMC



Avec technique de Pipeline



## LE CPU ET LA MÉMOIRE II

### PIPELINE

- Il ne s'agit pas d'augmenter la vitesse d'exécution d'une opération mais de produire davantage de résultats par seconde
- Le débit des résultats est environ  $N$  fois plus important dans le cas d'une unité opérationnelle divisée en  $N$  partie que pour l'unité non segmentée

#### À Gérer

- Problème pour les branchements
  - Pipelines séparés pour les deux possibilités
  - Prédiction basée sur le branchement effectué à l'exécution précédente
- Problème de dépendance entre instructions
  - réarrangement de la séquence d'instruction pour maintenir le pipeline plein



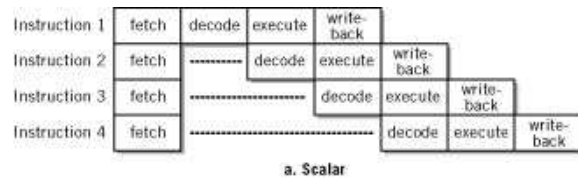
Complice la circuiterie électronique

## LE CPU ET LA MÉMOIRE II

### PROCESSEUR SCALAIRE/SUPER-SCALAIRE

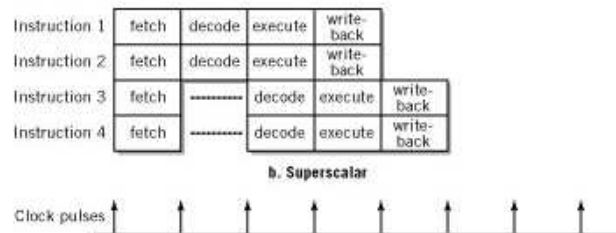
#### Processeur Scalaire

Processeur pour lequel la vitesse d'exécution moyenne d'une instruction égale approximativement la vitesse d'un cycle d'horloge



#### Processeur SuperScalaire

Nom donné à des processeurs RISC récents capables d'exécuter **en parallèle** un petit nombre d'instructions



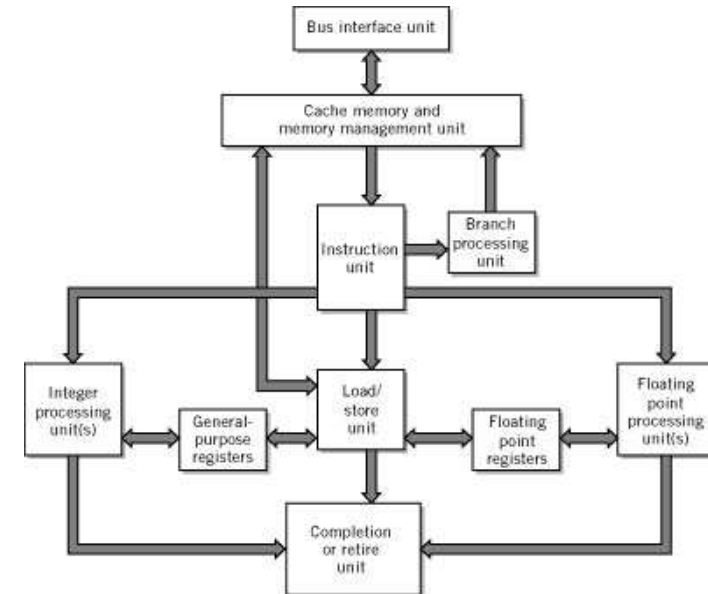
[Out-of-order Processing][Hazard]  
problème de dépendance à gérer

[Flow or Branch Dependencies]

## LE CPU ET LA MÉMOIRE II

### PROCESSEUR SUPERSCLAIRE

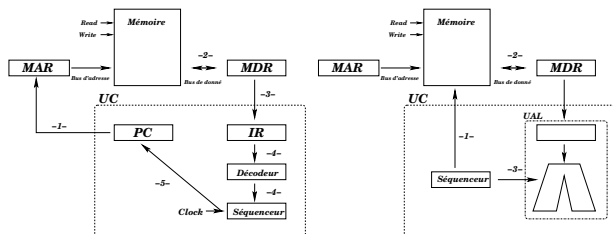
Bloc Diagramme du CPU superscalaire pipeline  
Pentium et Itanium



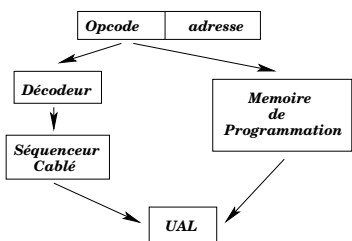
- L'unité d'instruction est responsable de maintenir le pipeline plein et pour dispatcher les instructions

## LE CPU ET LA MÉMOIRE II SÉQUENCEUR MICRO-PROGRAMMÉ

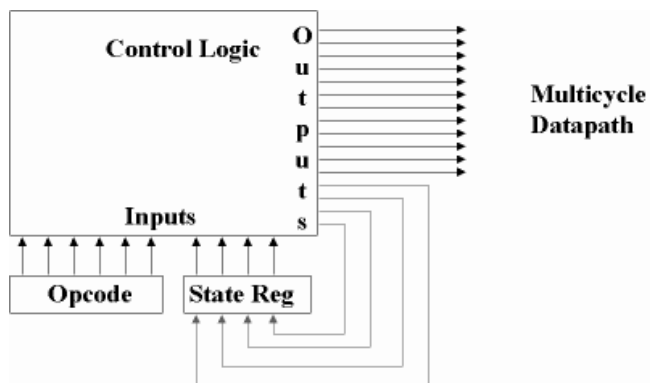
### Séquenceur



- **Séquenceur** : Automate générant les commandes nécessaires pour actionner/contrôler les unités participant à la recherche/exécution d'une instruction



### Séquenceur microprogrammé



## LE CPU ET LA MÉMOIRE II SÉQUENCEUR MICRO-PROGRAMMÉ

