Ch. 1
Introduction

B. Quoitin
(bruno.quoitin@umons.ac.be)

Introduction

- Combien de processeurs avez-vous à la maison ?

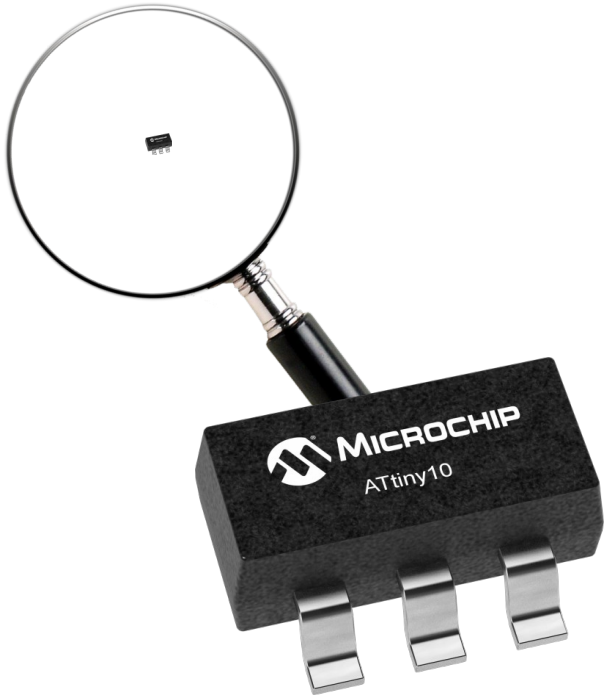
Ordinateurs

Smartphones, tablettes, consoles, téléviseurs

Systèmes embarqués

Introduction

"*Les ordinateurs utilisent à peu près **0 %** de l'ensemble des processeurs du monde.*"
(EE|Times, *Embedded Processors by the Numbers*, Jim Turley, Jan. 1999)

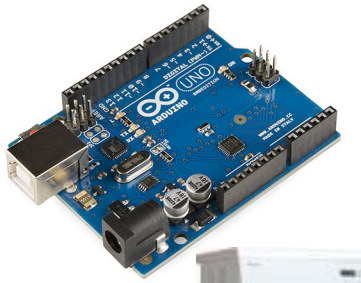


Classification des Ordinateurs

- **Généralistes** – *polyvalents*
  - Traitement de texte, tableur, multimedia, jeux, applications scientifiques, ...
  - Compromis coût / performance

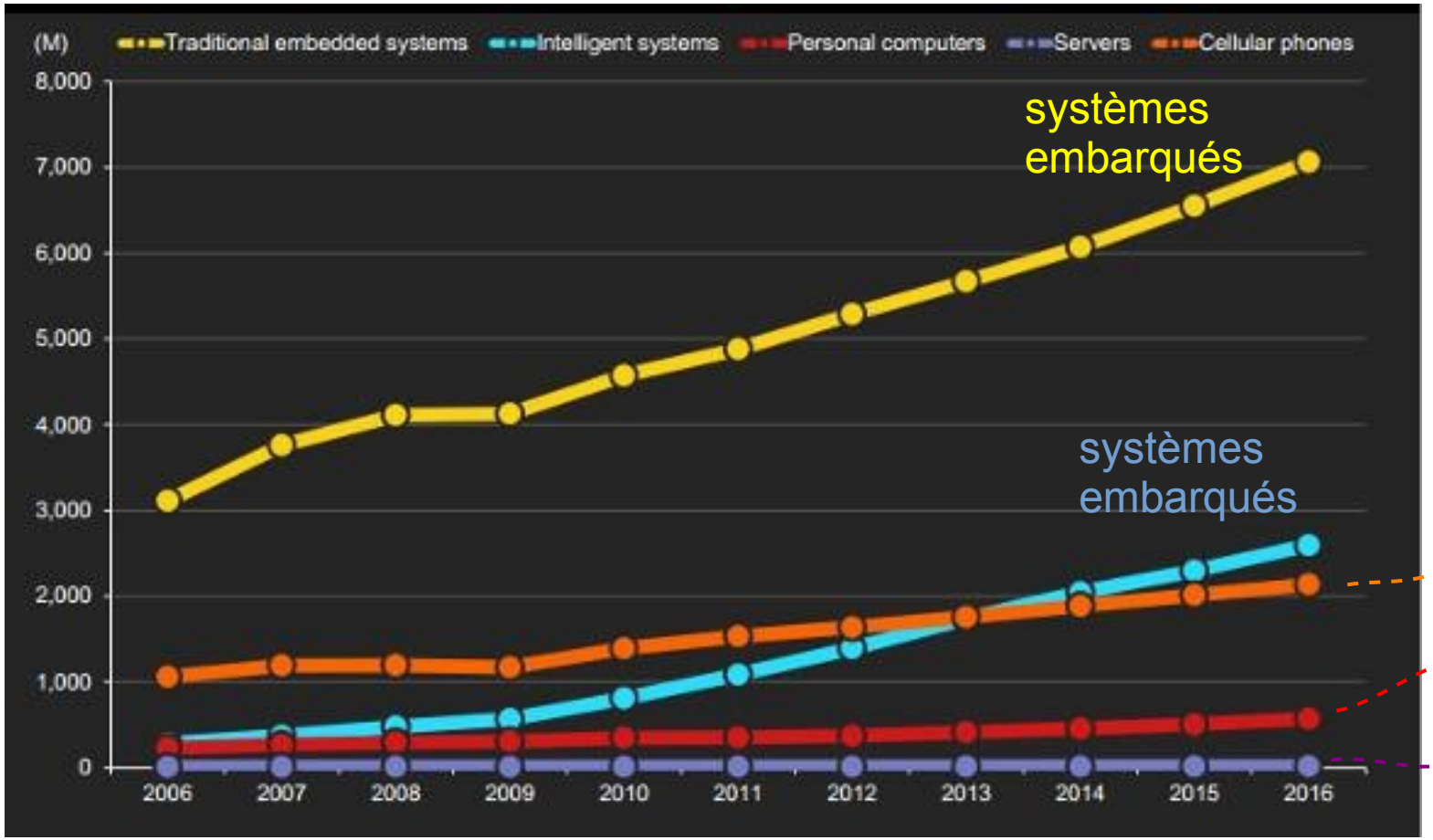
- **Serveurs** – *spécialisés, partagés*
  - Bases de données, serveurs de streaming, calcul haute-performance (*HPC*), ...
  - Haute capacité / disponibilité / performance / robustesse → coût généralement plus élevé

- **Systèmes Embarqués** – *très spécialisés*
  - Conçus pour une application unique (e.g. contrôle d'une machine à laver, de la téléphonie, ...)
  - Compromis coût / performance / consommation énergétique
  - Fiabilité (ABS voiture, fusée ARIANE-5) !!



Importance des Systèmes Embarqués

- **Marché des Ordinateurs**
  - Les processeurs les plus vendus sont ceux destinés aux systèmes embarqués !



Source : http://www.theregister.co.uk, 23/04/2013

Table des Matières

Modèle d'un Ordinateur

- **Que contient un ordinateur ?**
  - Quelque soit son type, **les principes de fonctionnement** sous-jacents à un ordinateurs **sont les mêmes**.
  - Chaque ordinateur comprend les composants suivants:
  - **Processeur**
    - Interprète et exécute les instructions des programmes
  - **Mémoire**
    - Stocke des données ET des programmes (modèle parfois dit « de von Neuman »)
  - **Horloge**
    - Détermine le rythme d'exécution des opérations
  - **Entrées/sorties**

Composantes d'un Ordinateur

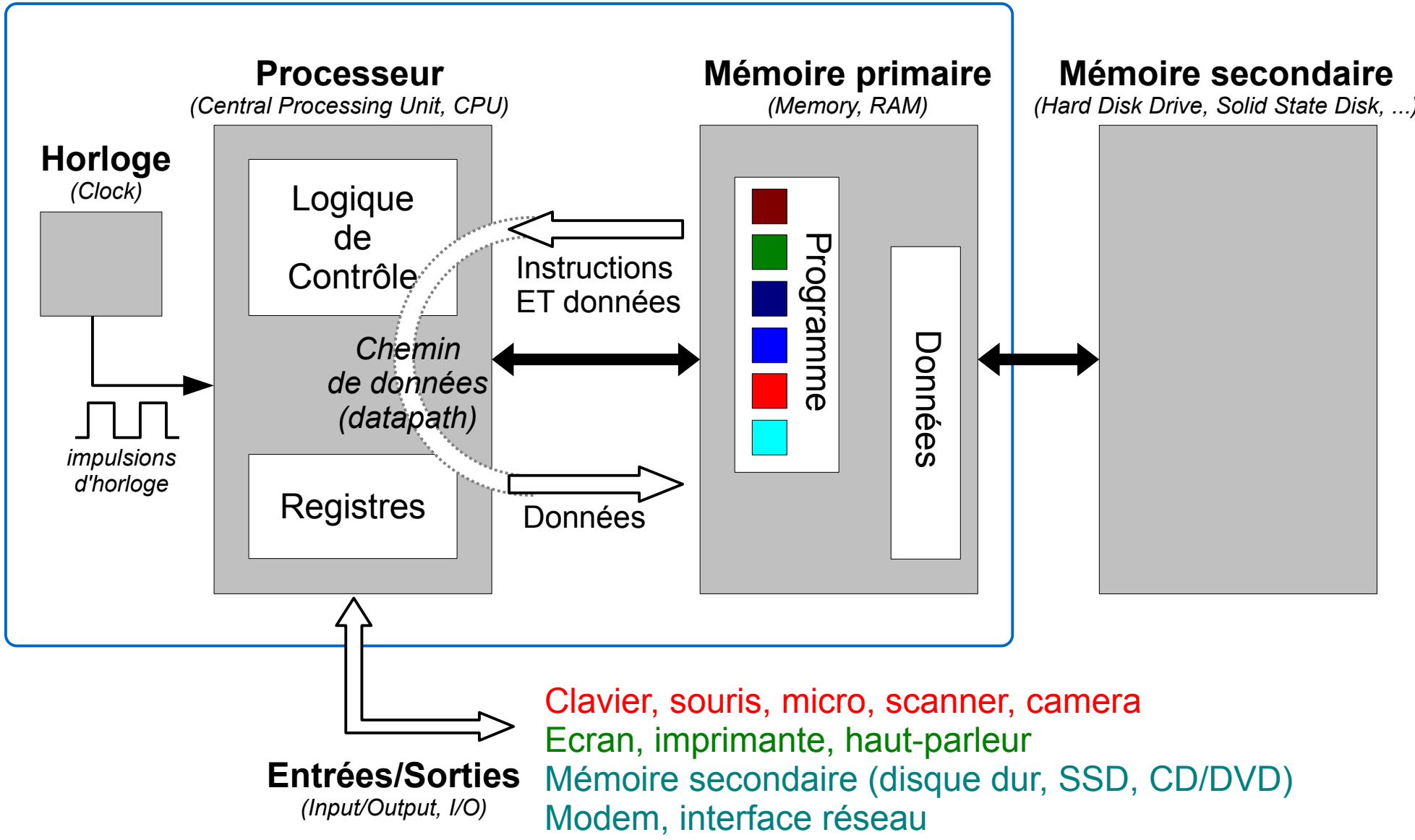


Table des Matières

Exécution d'un Programme

- **Exécution d'un programme**
  - Le processeur exécute des opérations élémentaires
    - transférer une donnée de la mémoire vers un registre
    - calculer des opérations arithmétiques avec des registres
    - transférer une donnée d'un registre vers la mémoire
    - …
  - Le processeur est une implémentation (matérielle) d'un algorithme qui effectue en boucle les opérations suivantes

(1) lire une instruction en mémoire
(2) déterminer l'opération à effectuer
(3) exécuter l'opération
(4) passer à l'instruction suivante

} = *cycle processeur*

L'exécution de cette boucle est rythmée par l'horloge du processeur.

Exécution d'un Programme

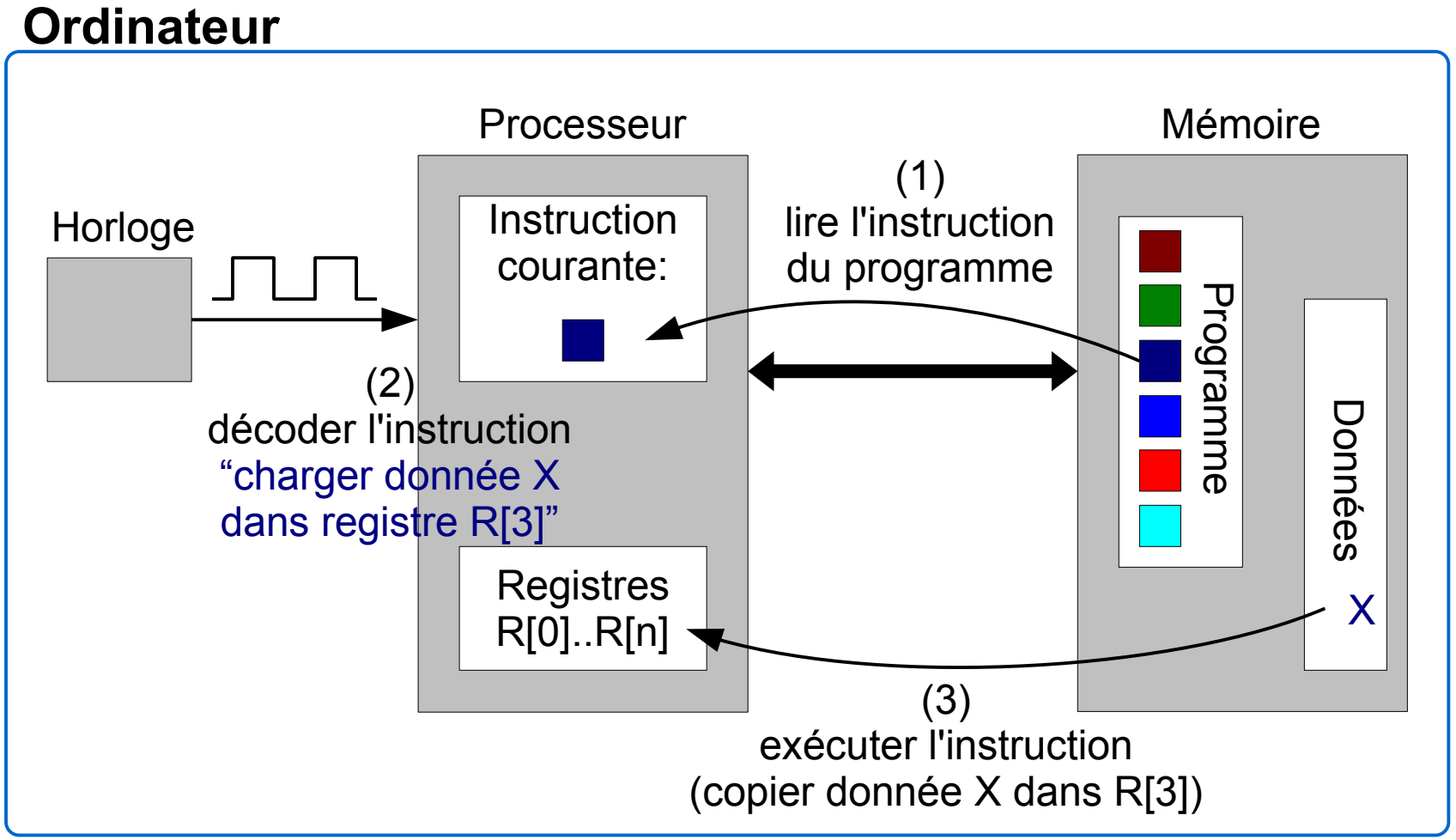


Table des Matières

Interface Logiciel / Matériel

- **Architecture**
  - Pour pouvoir exécuter un programme sur un processeur donné, il faut en respecter l'architecture.
  - ***Instruction Set Architecture*** (ISA) ou **Architecture**
    - Modèle abstrait d'un processeur renseignant tout ce qu'il est nécessaire de connaître pour réaliser un programme fonctionnel (registres, jeu d'instructions, organisation des entrées/sorties, accès à la mémoire, interruptions, exceptions, etc.)
    - Exemples: x86, PowerPC, ARM, MIPS, RISC-V

      "CISC" "RISC"
  - **Jeu d'Instructions** (*instruction set*)
    - Chaque architecture définit un *jeu d'instructions* que le processeur peut exécuter.
    - Exemples d'instructions MIPS : `add` r_0, r_1, r_2 ; `load` r_0, *offs*(*base*)

Interface Logiciel / Matériel

- **Abstraction du matériel**
  - Comment **simplifier/accélérer** la conception de programmes ?
  - Comment en **augmenter la portabilité**(1) ?

  - **Cacher les détails matériels du système**
    - utilisation de langages de haut-niveau (C, Java, Python, …)
    - utilisation d'outils spécialisés pour traduire les programmes en instructions du processeur.
    - définir des interfaces claires entre matériel et logiciel

  - **Déléguer la gestion des ressources du système**
    - ressources = mémoire, périphériques, processus
    - logiciels système, notamment le **système d'exploitation** (*operating system - OS*).

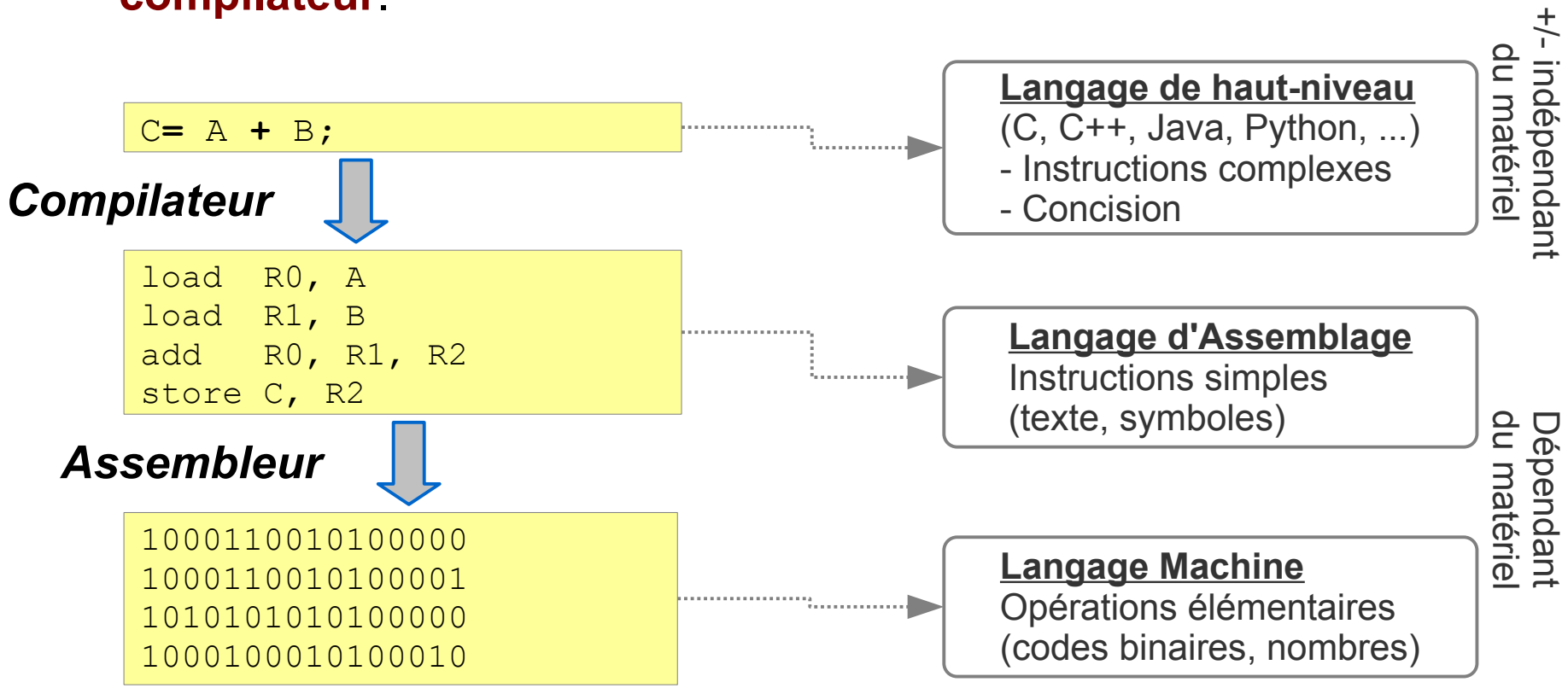
(1) permettre à une application de fonctionner sur des systèmes différents.

Interface Logiciel / Matériel

- **Langages de haut-niveau**
  - Langage machine (bas niveau)
    - Très détaillé
    - Dépendant (de l'architecture) du processeur
    - Permet seulement d'exprimer des **opérations élémentaires** (opérations arithmétiques, déplacement de données simples en mémoire, etc.)
  - Langages de haut-niveau
    - Permettent d'**exprimer des opérations plus complexes** que les opérations élémentaires supportées par le processeur.
    - **Cachent certains détails de l'implémentation matérielle** de l'ordinateur (p.ex. l'emplacement d'une variable en mémoire).
    - Exemples : C, C++, Rust, Java, C#, Python, ...

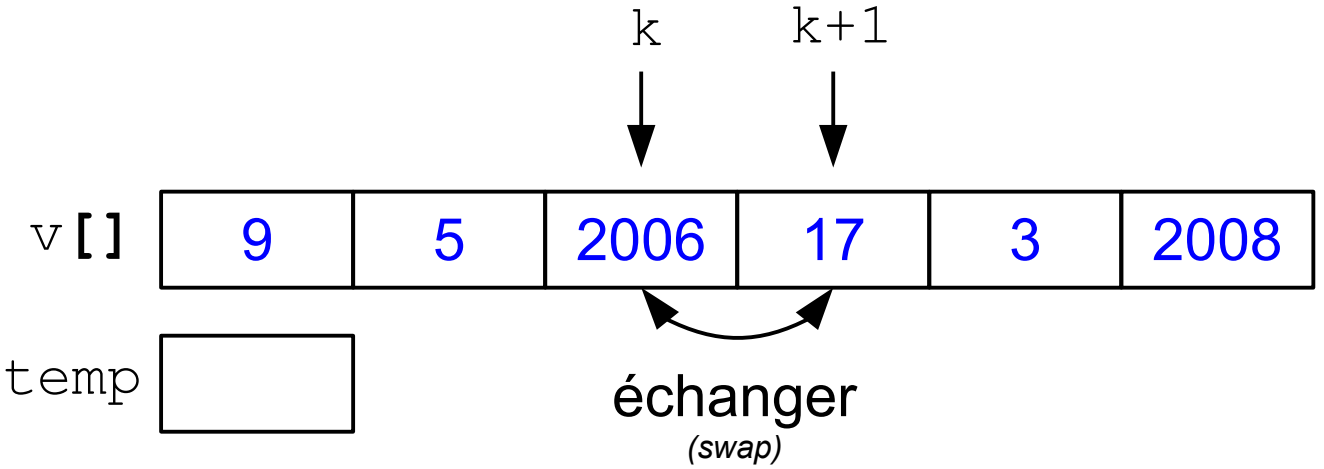
Interface Logiciel / Matériel

- **Langages de haut-niveau**
  - Compilation
    - Mécanisme pour traduire les opérations d'un langage de haut-niveau en une suite d'instructions élémentaires pour un processeur.
    - La compilation est généralement effectuée par un programme : le **compilateur**.



Interface Logiciel / Matériel

- **Langages de haut-niveau**
  - Exemple en langage C
    - Echanger le contenu de 2 cellules consécutives d'un tableau d'entiers

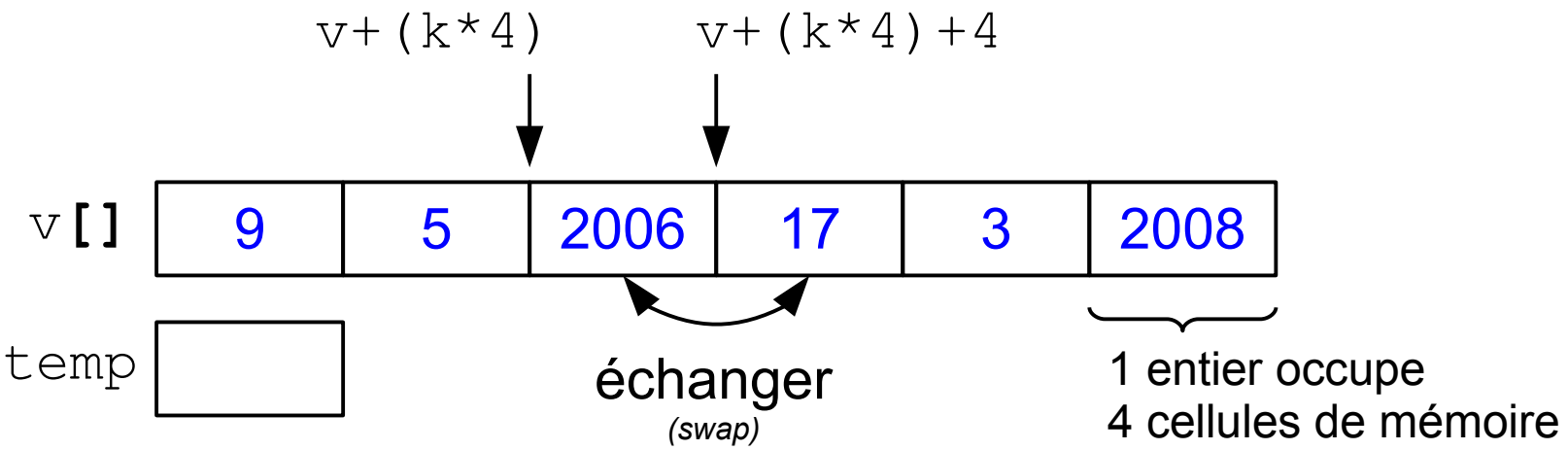



```
void swap(int v[], int k)
{ int temp;
  temp= v[k];
  v[k]= v[k+1];
  v[k+1]= temp;
}
```

Interface Logiciel / Matériel

- **Langages de haut-niveau**
  - Exemple en langage C

A ce stade, vous ne devez pas comprendre les détails de cet exemple.
Ces détails seront expliqués dans un chapitre ultérieur.



Allocation de registres:
R[5] = argument `v[]`
(position de `v[]` en mémoire)
R[4] = argument `k`
R[2], R[15], R[16] =
registres temporaires

Opérations élémentaires
1). R[2] = R[5] * 4
2). R[2] = R[2] + R[4]
3). R[15] = charger(R[2]+0)
4). R[16] = charger(R[2]+4)
5). stocker(R[2]+0, R[16])
6). stocker(R[2]+4, R[15])

Interface Logiciel / Matériel

- **Langages de haut-niveau**
  - Exemple en langage C

A ce stade, vous ne devez pas comprendre les détails de cet exemple.
Ces détails seront expliqués dans un chapitre ultérieur.

```
void swap(int v[], int k)
{ int temp;
  temp= v[k];
  v[k]= v[k+1];
  v[k+1]= temp;
}
```

(langage C)

Compilateur

```
swap:
   muli $2, $5, 4
   add  $2, $4, $2
   lw   $15, 0($2)
   lw   $16, 4($2)
   sw   $16, 0($2)
   sw   $15, 4
   jr   $31
```

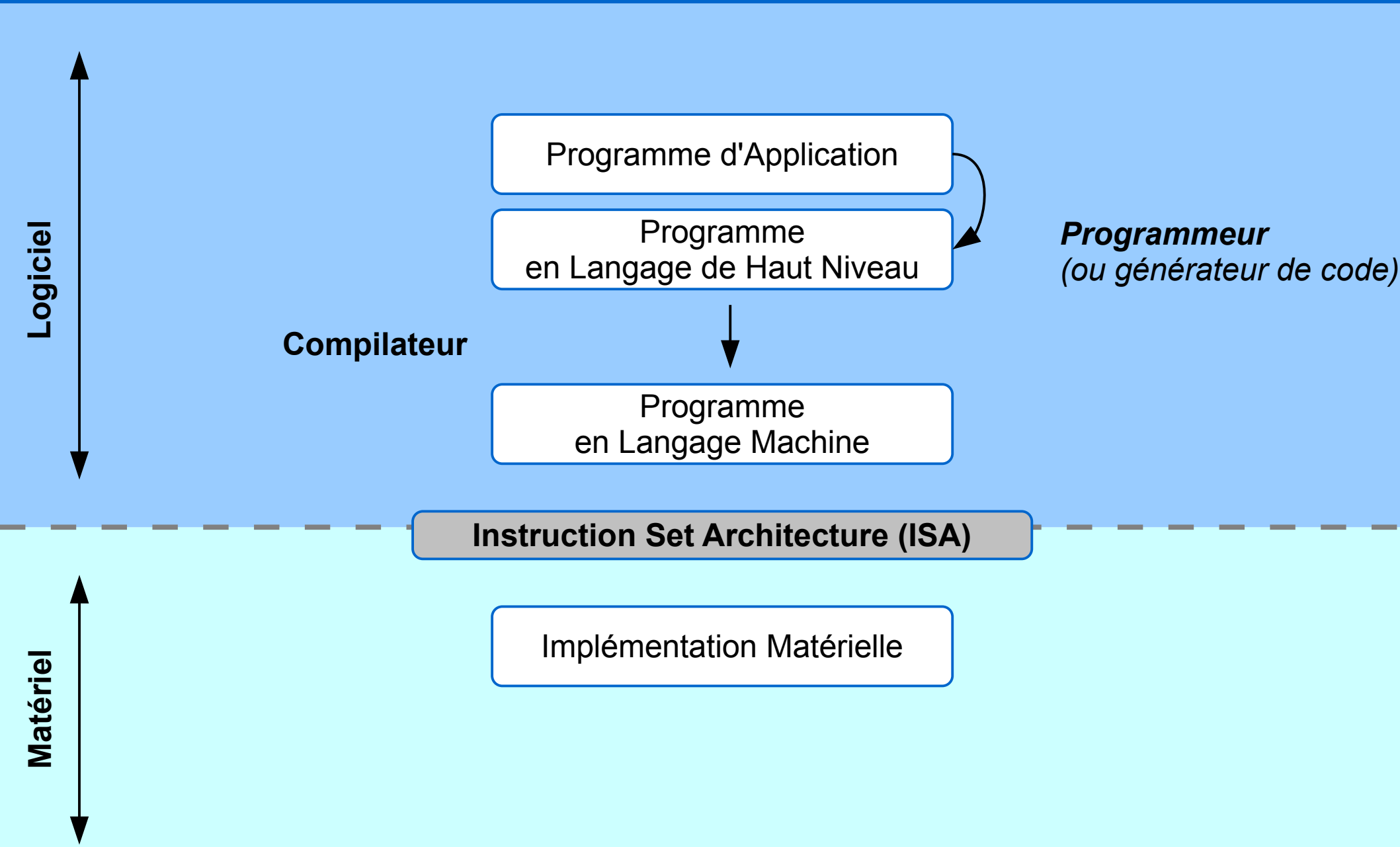
(langage d'assemblage MIPS)

Assembleur

```
00000000101000010000000000011000
00000000000011000000110000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

(instructions processeur architecture MIPS)

Interface Logiciel / Matériel



Interface Logiciel / Matériel

Logiciel

Programme d'Application

Programme
en Langage **C**

Compilateur C **x86**

Programme
en Langage Machine **x86**

Architecture **x86**

Matériel

Implémentation Matérielle
Intel x86

Interface Logiciel / Matériel

Logiciel

Programme d'Application

Programme
en Langage **C**

Compilateur C **x86**

Programme
en Langage Machine **x86**

Compatibilité binaire des programmes

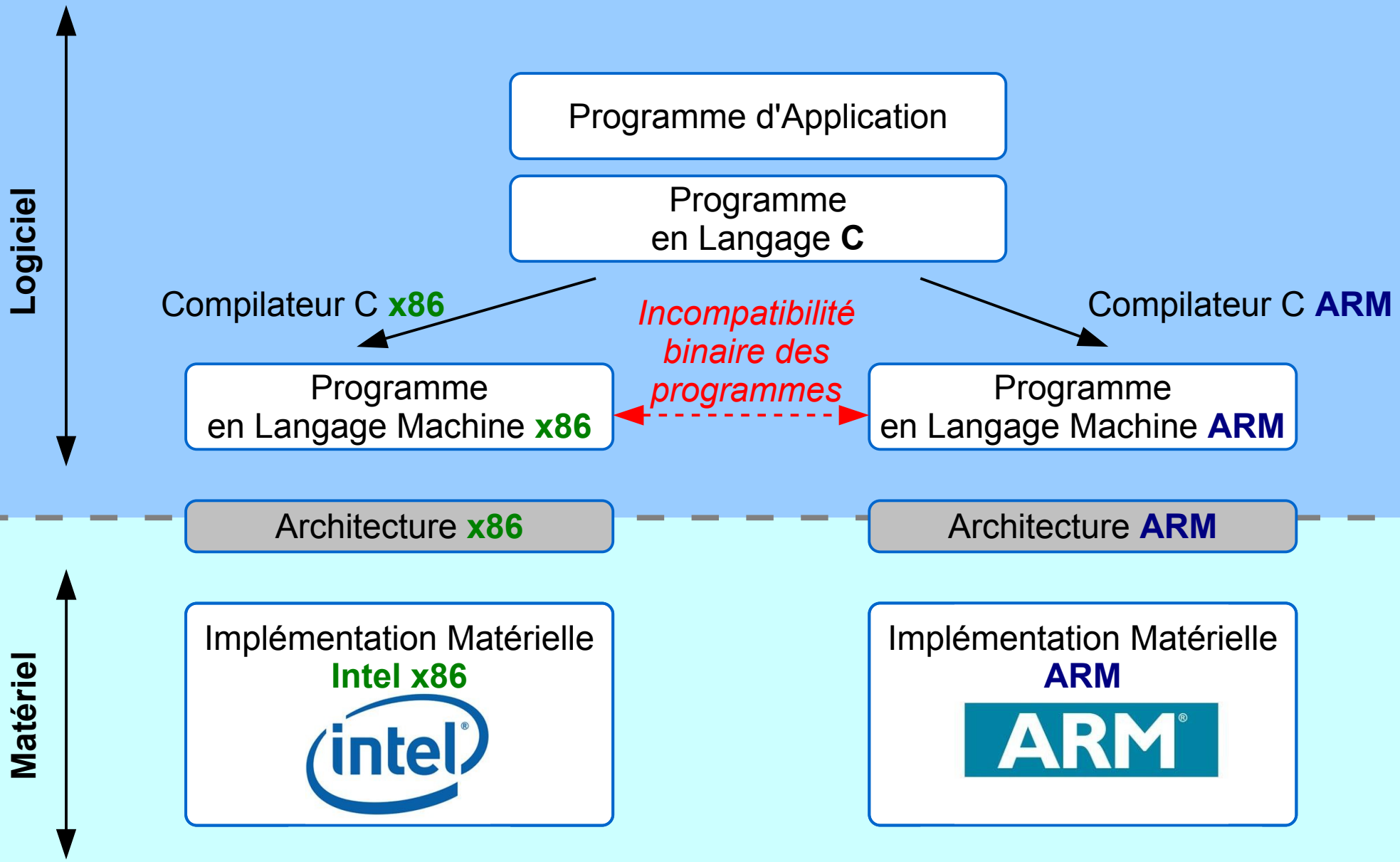
Architecture **x86**

Matériel

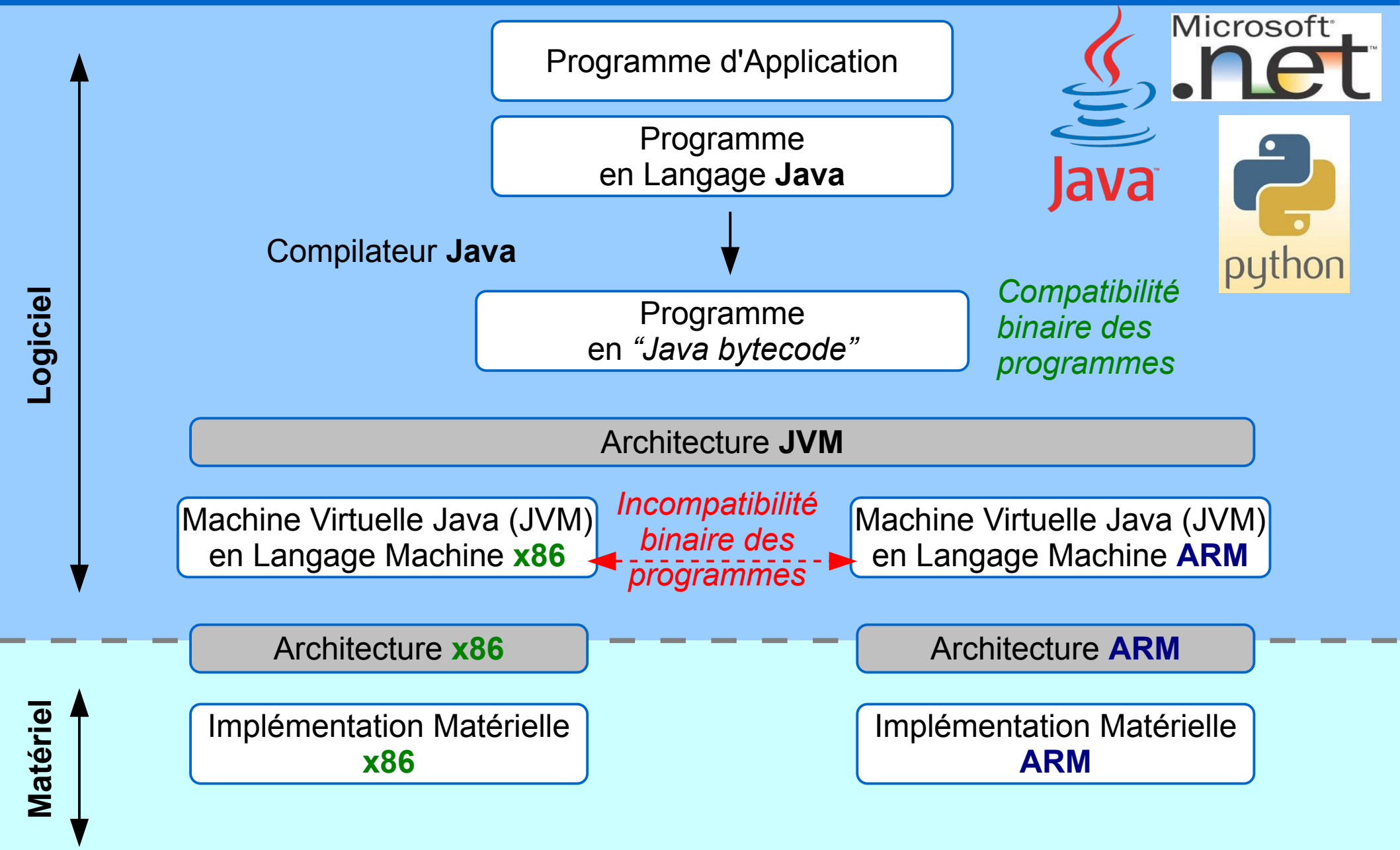
Implémentation Matérielle
Intel x86

Implémentation Matérielle
AMD x86

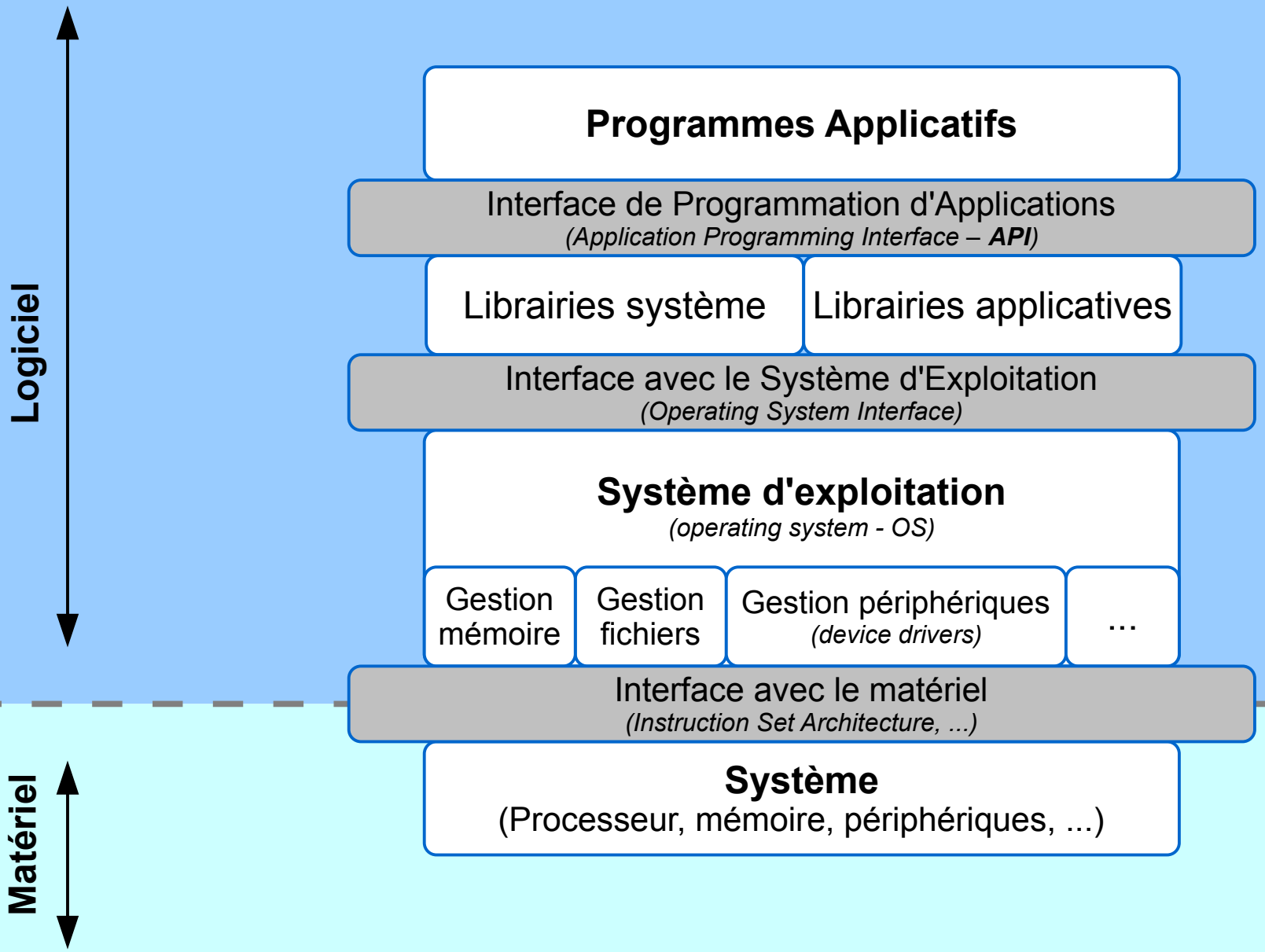
Interface Logiciel / Matériel



Machine Virtuelle Java & co



Interface Logiciel / Matériel



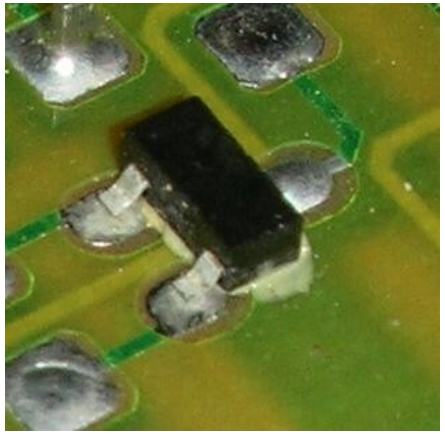
- abstrait les ressources matérielles en fournissant une interface simplifiée.
- gère l'accès de plusieurs programmes aux ressources du système.

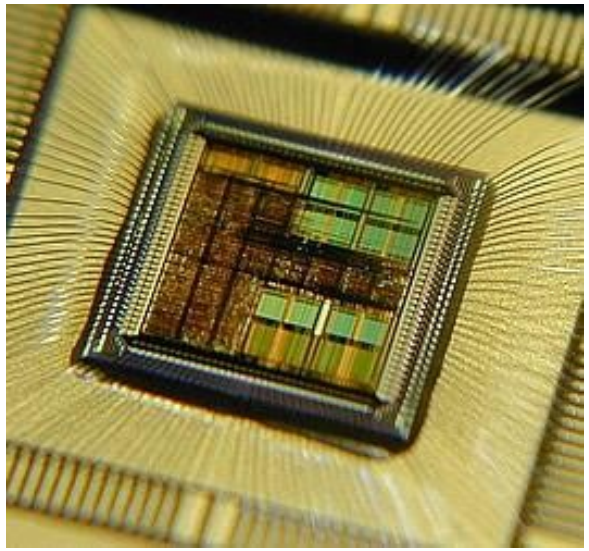
Table des Matières

Densité des Composants

Year	Technology	Performance relative / coût unitaire
1951	tubes	1
1965	transistors	35
1975	circuits intégrés (MSI)	900
1995	VLSI	2 400 000
2005	ULSI	6 200 000 000

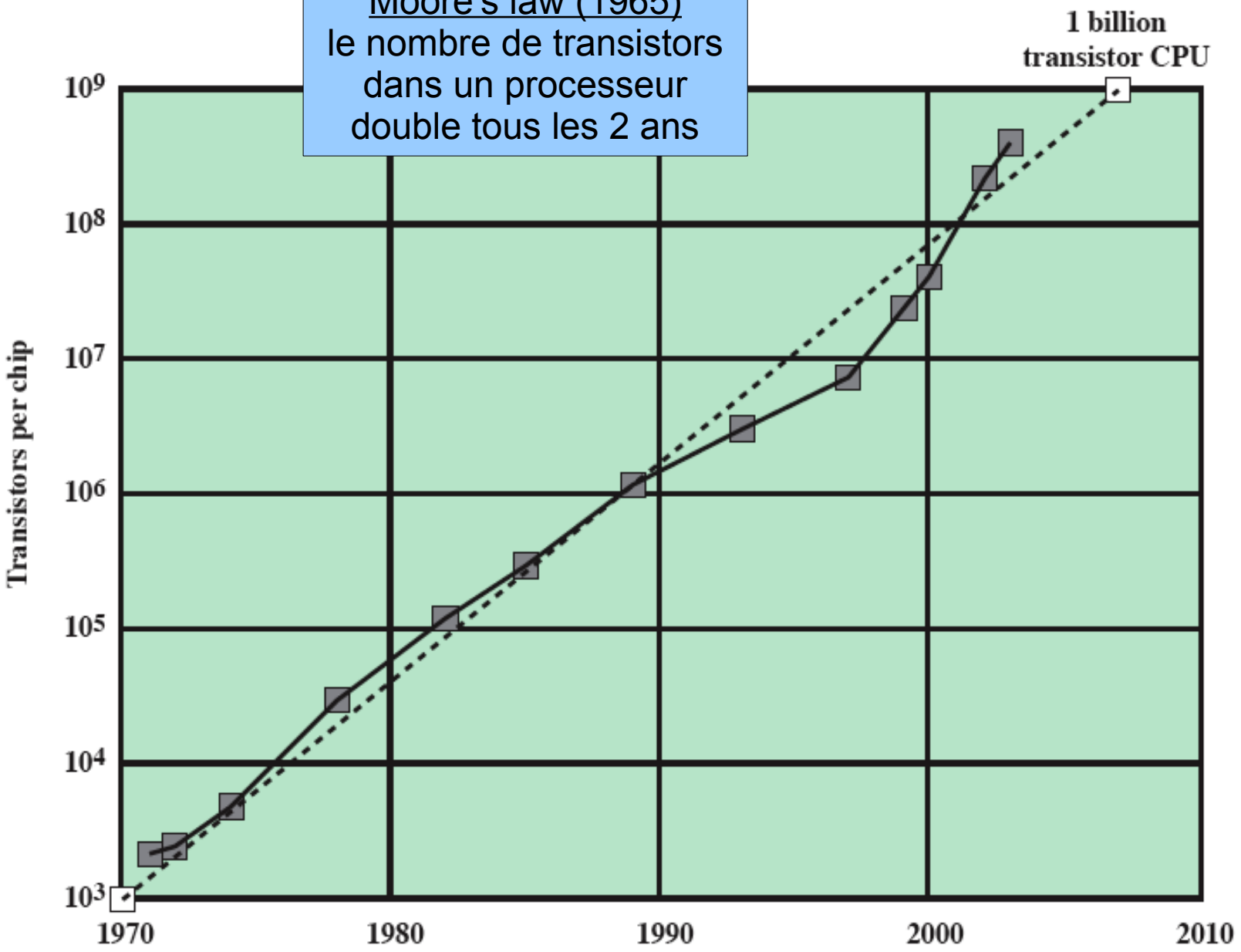
Source: Computer Organization and Design, Patterson, MK, 2008





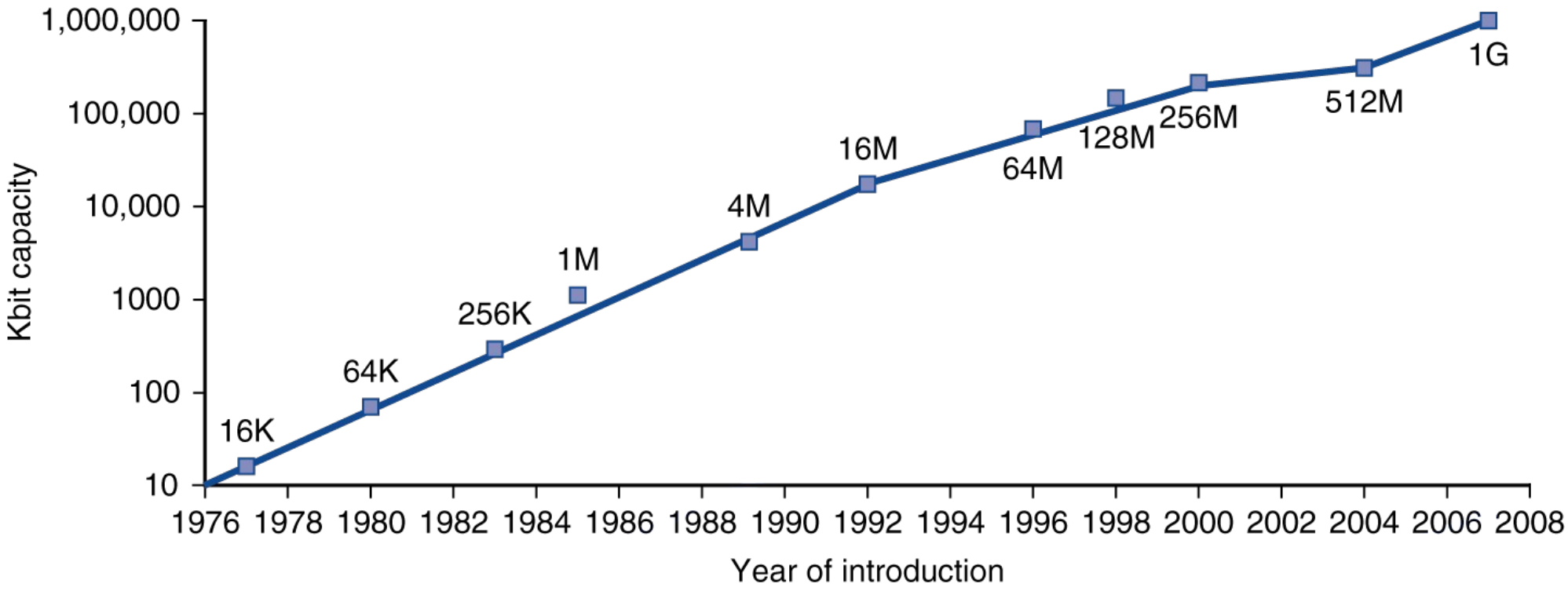
Nombre de transistors / CPU

Moore's law (1965)
le nombre de transistors dans un processeur double tous les 2 ans



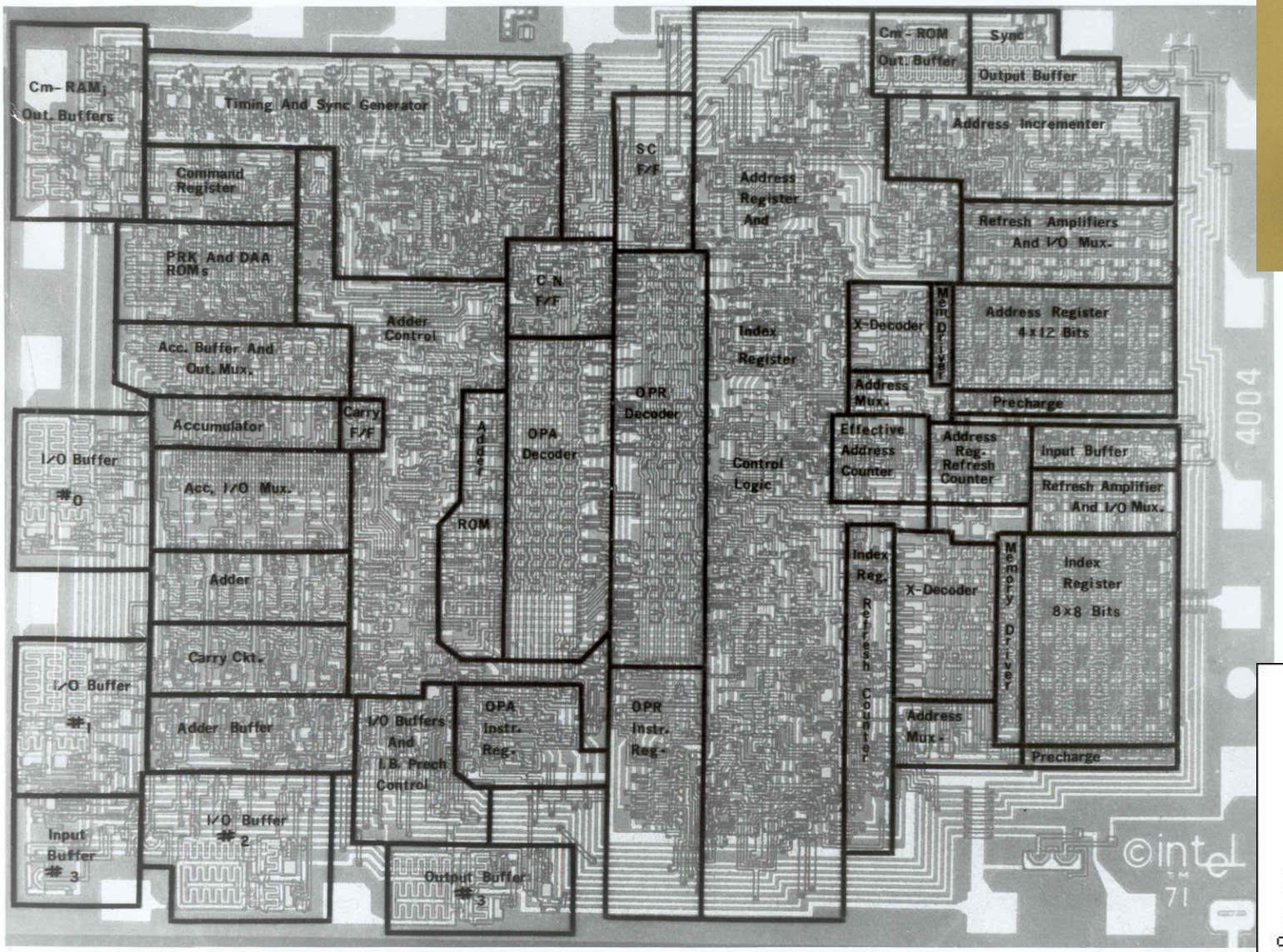
Source:
Computer Organization and Architecture
Stallings, Pearson, 2010

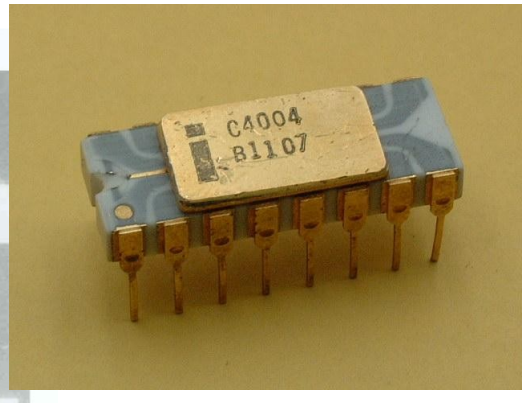
Capacité des DRAMs



Source: Computer Organization and Design, Patterson, MK, 2008

Intel 4004 (1971)

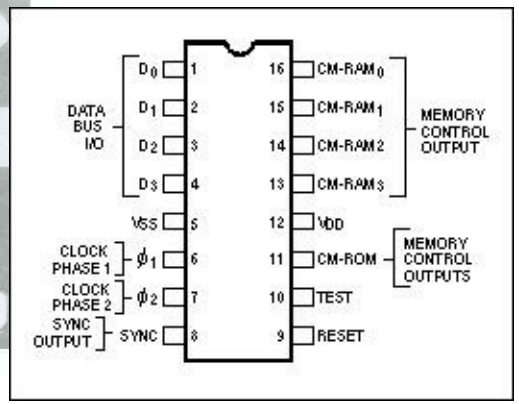




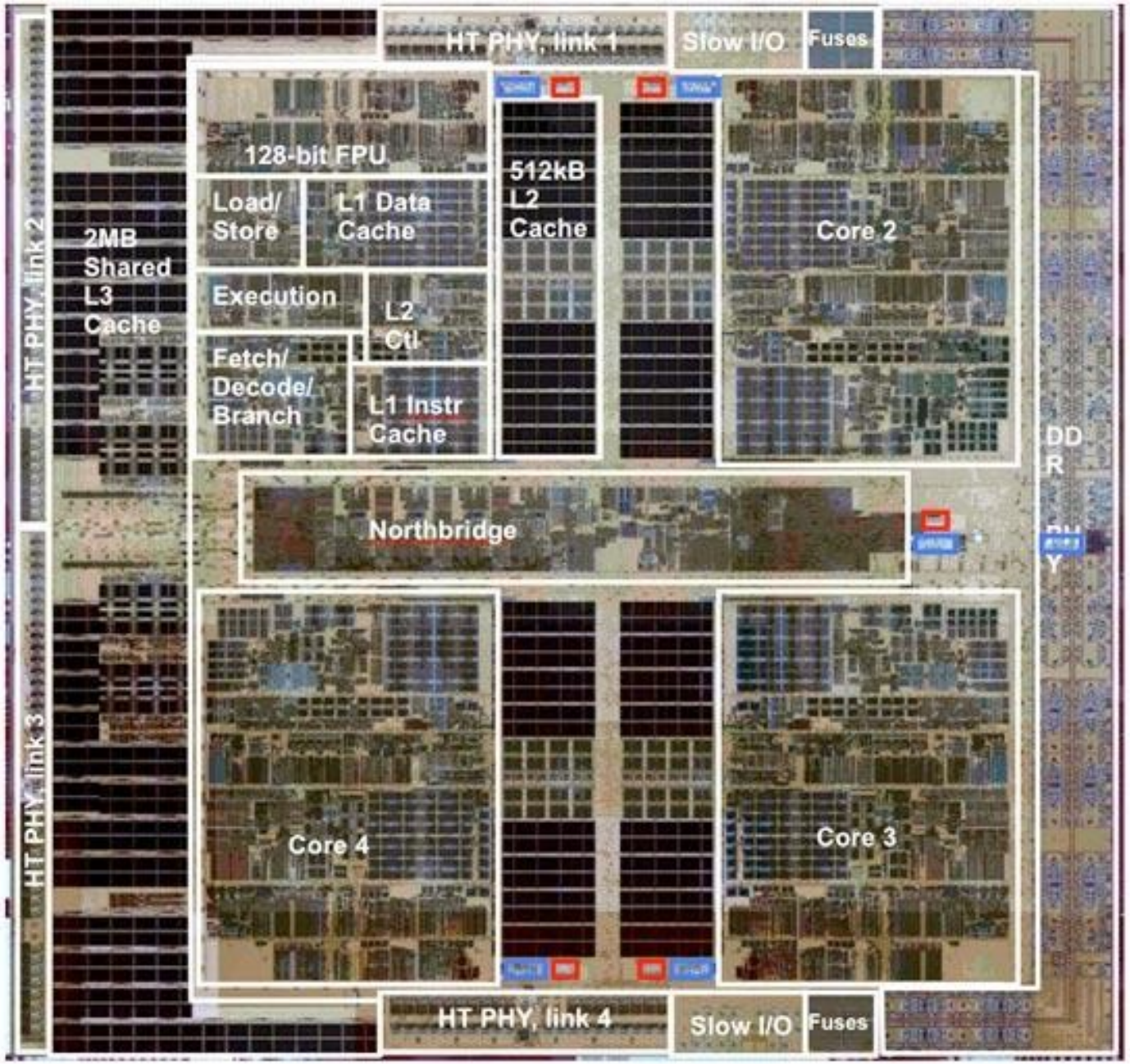
24 mm^2

2300 transistors

16 broches



AMD Barcelona (2007)



285mm^2

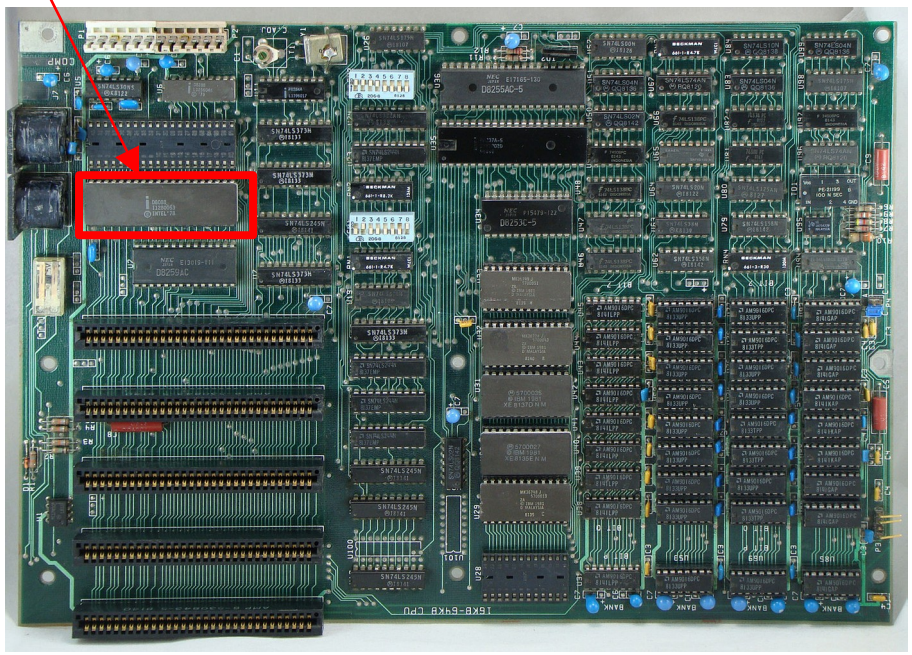
463M transistors

1207 broches

95 Watts

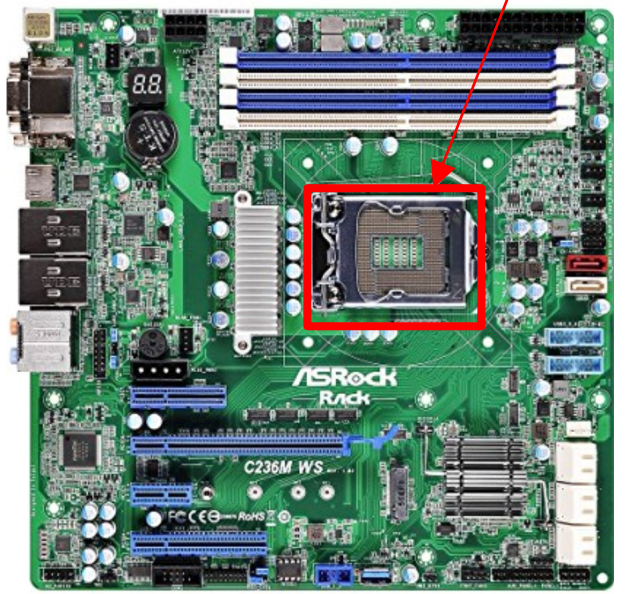
Densité des Composants

Processeur



IBM PC AT motherboard
(Intel 8088 processor)

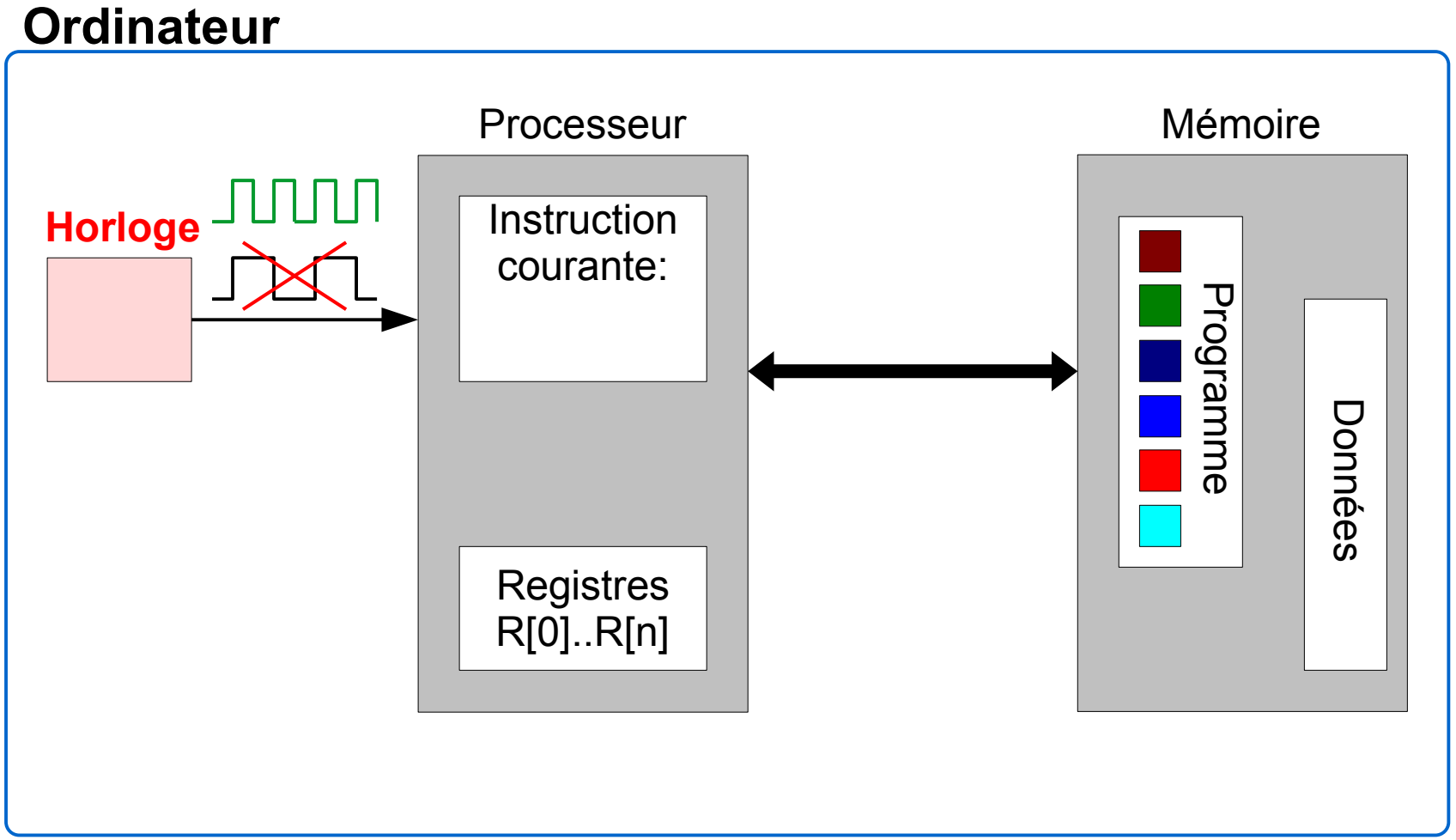
(Place du)
Processeur



Modern PC motherboard
(Intel Xeon)

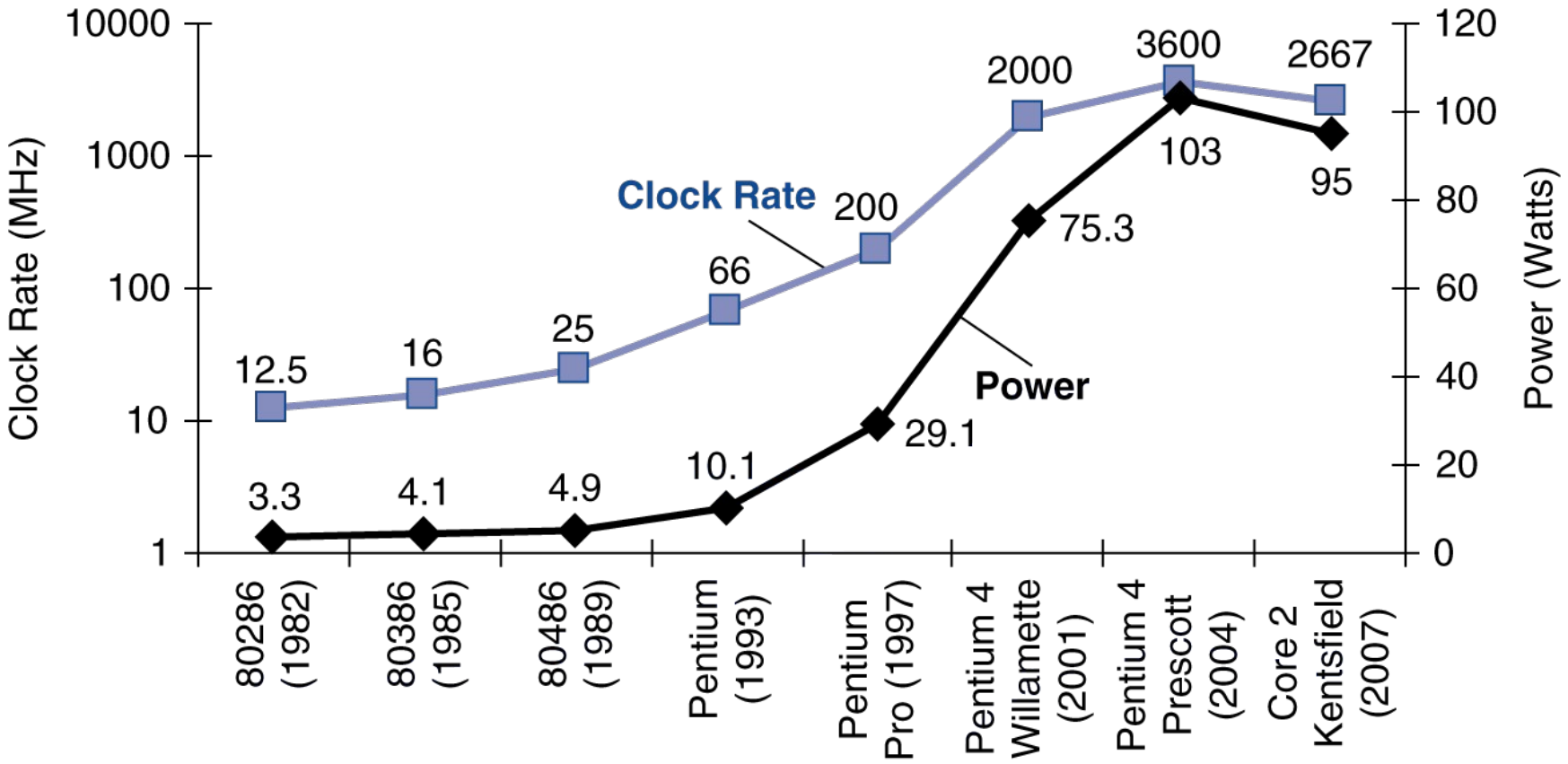
Les processeurs intègrent de plus en plus de fonctions dans un seul “chip”. A gauche, le processeur est entouré de nombreux autres circuits intégrés (“chip set”). A droite, presque tout est intégré au processeur.

Augmentation de la fréquence



Accroître la fréquence d'horloge permet d'augmenter la vitesse de traitement d'un système informatique. Cette augmentation n'est pas illimitée.

Limite de Puissance

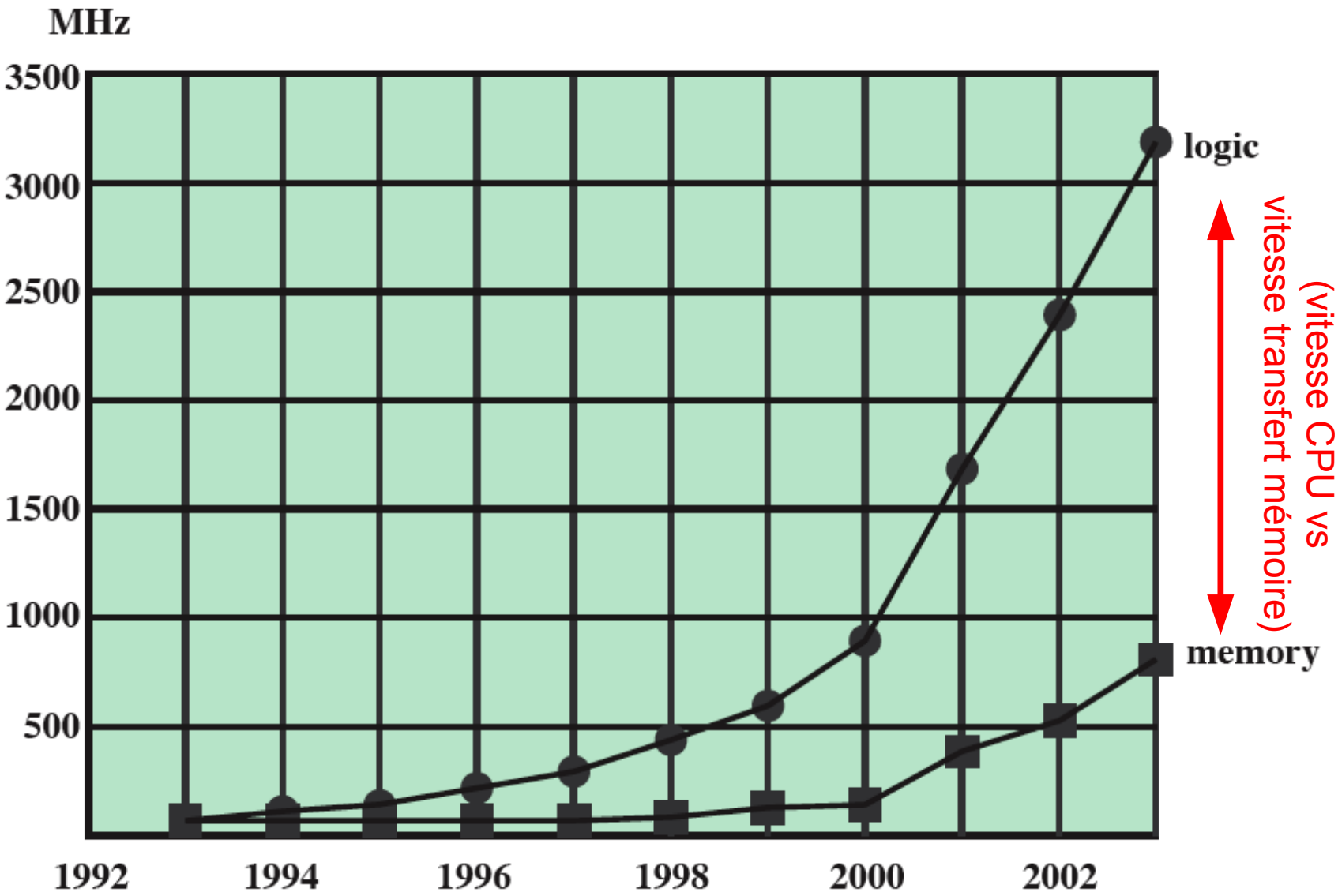


Source : Computer Organization and Design, Patterson, MK, 2008

Augmentation fréquence horloge + augmentation densité transistors
=
Augmentation puissance dissipée / cm^2

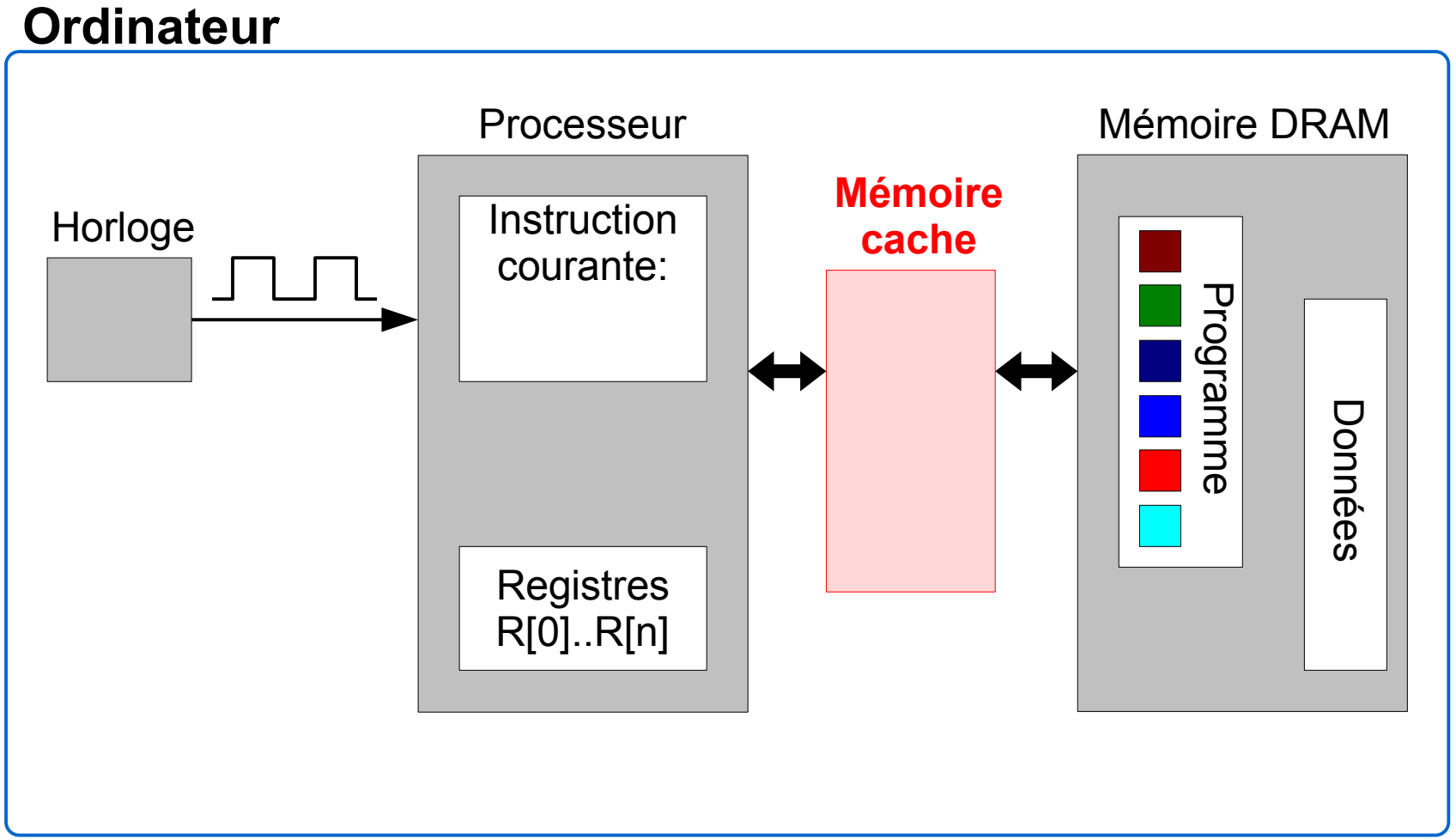
Refroidissement nécessite techniques trop coûteuses pour le marché des ordinateurs personnels !!!

Déséquilibre CPU / Mémoire



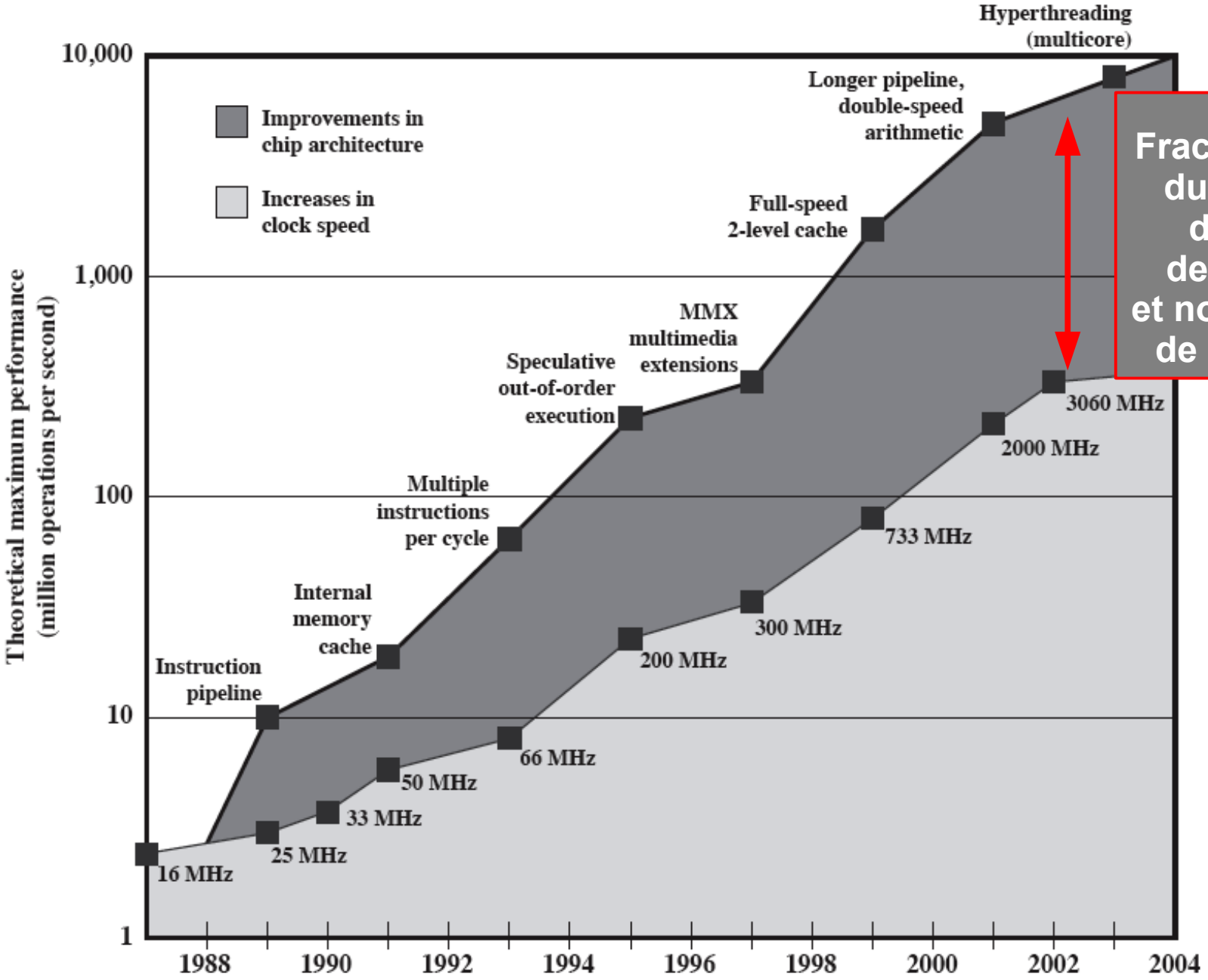
Source:
Computer Organization and Architecture
Stallings, Pearson, 2010

Déséquilibre CPU / Mémoire



L'introduction d'une mémoire cache permet d'amenuiser l'impact de la différence de "vitesse" entre processeur et mémoire DRAM.

Complexité de l'Architecture des CPUs



Fraction importante du gain provient de l'évolution de l'architecture et non de l'élévation de la fréquence !!

Source:
Computer Organization and Architecture
Stallings, Pearson, 2010