Cours microprocesseurs

Denis Barthou

dbarthou@enseirb-matmeca.fr

Plan

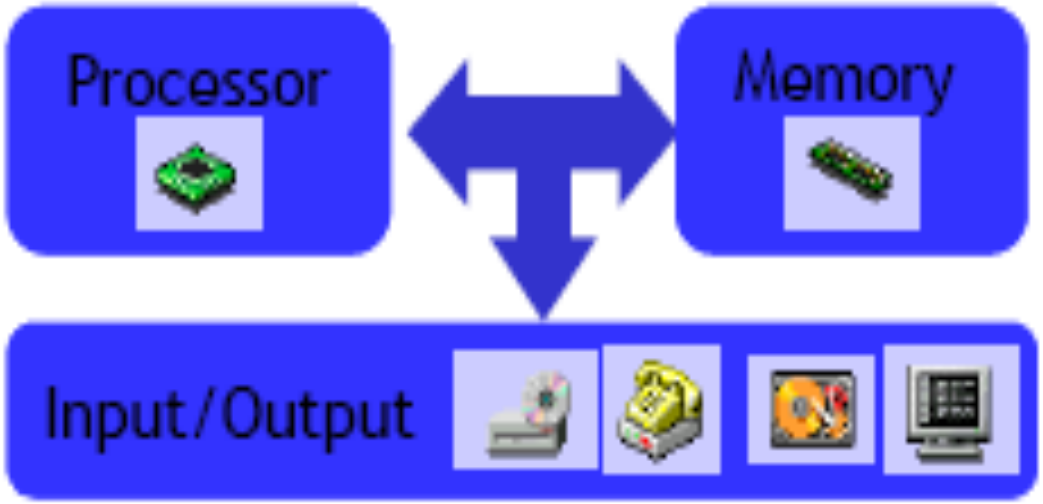
1. *Introduction, contexte*
2. Codage des nombres
3. Exécution simple des instructions
4. Pipeline
5. Système mémoire
6. Caches
7. Entrées/Sorties

1- Introduction

- Analyser et comprendre le fonctionnement d'ordinateur
- Lien entre matériel et logiciel
- Mécanismes clés: ISA, mémoire, pipeline, parallélisme

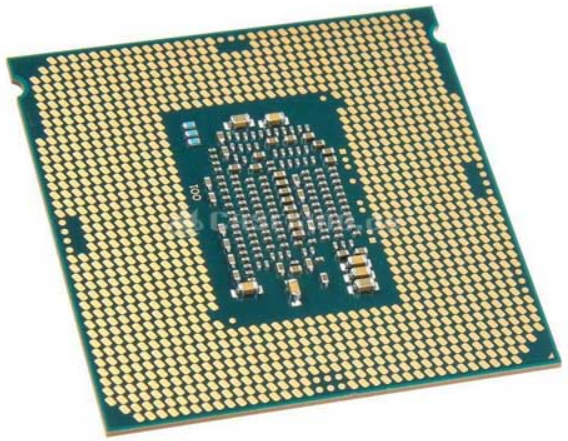
1- Introduction

- Analyser et comprendre le fonctionnement d'ordinateur
- Lien entre matériel et logiciel
- Principes clés: programmation assembleur, structure d'un ordinateur, processeur, mémoire, pipeline, parallélisme



Le processeur

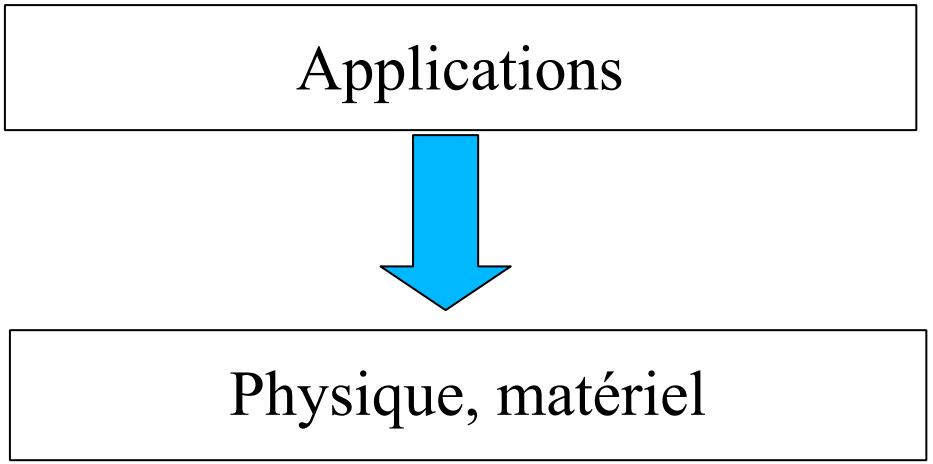
- L'unité centrale de la machine
  - Exécute les instructions (est programmable)
  - Effectue les calculs arithmétiques, des tests
  - Peut manipuler des données (des nombres) avec ses instructions
  - Peut accéder à une mémoire
  - Est cadencé par une horloge



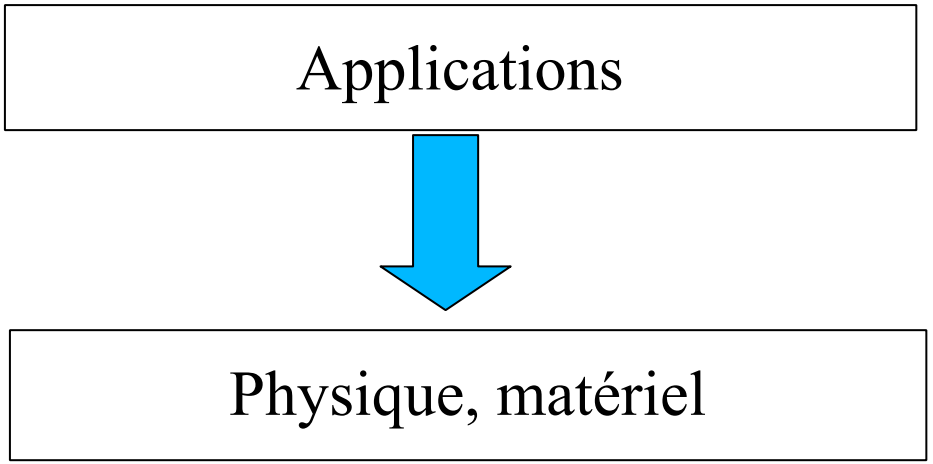
La mémoire

- Circuit qui permet de stocker des données et le programme
  - Peuvent être réécrites (RAM) ou pas (ROM)
  - S'efface ou pas, suivant la technologie utilisée, lorsqu'on éteint la machine

Architecture des ordinateurs

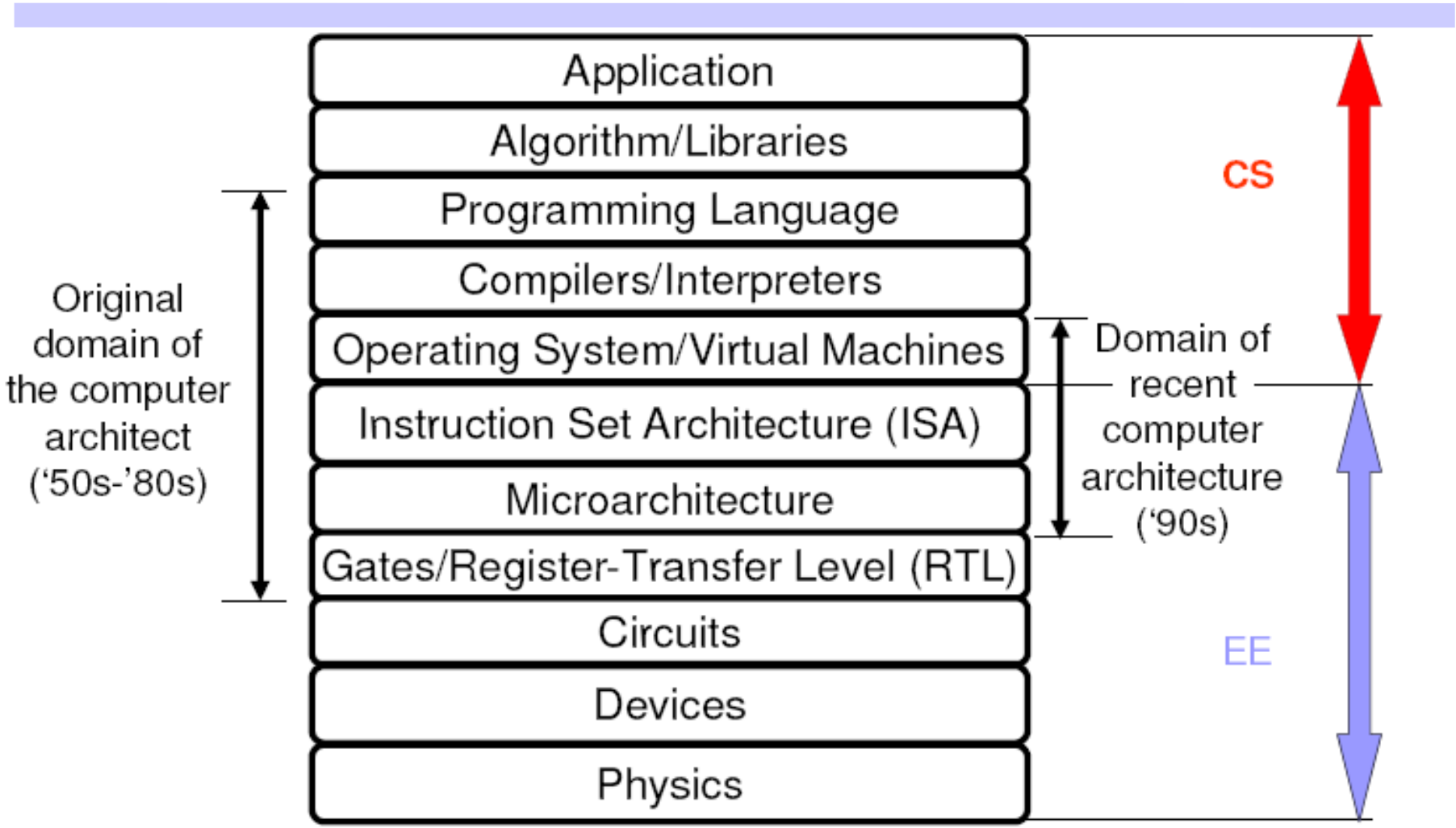


Architecture des ordinateurs



- **Ecart trop important** pour une seule étape
- Architecture des ordinateurs: au sens large, définition des étapes intermédiaires pour permettre la réalisation automatique de traitement de l'information

Couches d'abstractions



Un peu d'histoire

- 1642: Pascal, machine à calculer mécanique
- 1833, Babbage: instructions et conditionnelles sur cartes perforées
- 1890, Hollerith: machine pour bureau de recencement, fondation IBM
- 1936, Turing: formalisation d'un ordinateur; These de Church: tout algorithme peut être implémenté dans un programme de machine de Turing.
- 1945, Von Neumann: données et programmes en mémoire, modèle machine séquentielle

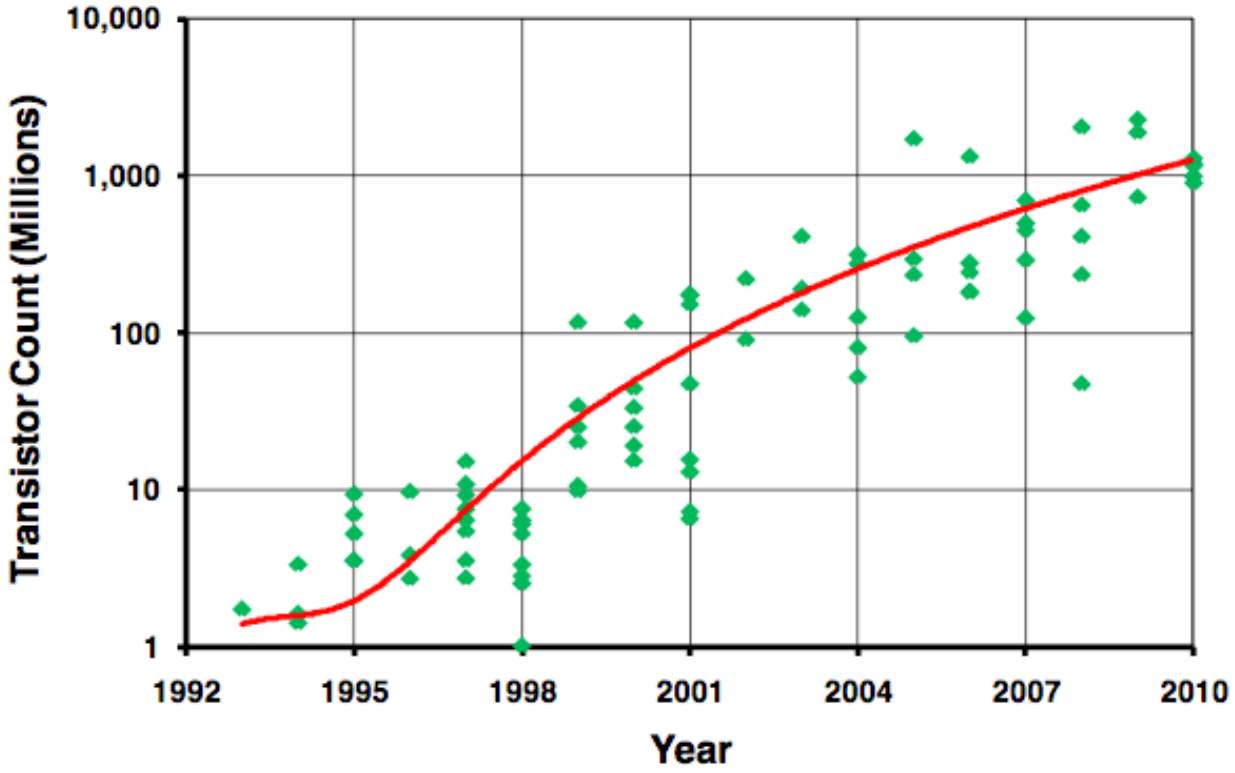
Eniac, 5000 additions/s, 30t, 150kW
(Wikipedia)

Un peu d'histoire

- 1947: invention transistor (Bells lab)
- 1958: invention circuit intégré (Texas Instrument)
- 1971: premier microprocesseur (Intel)

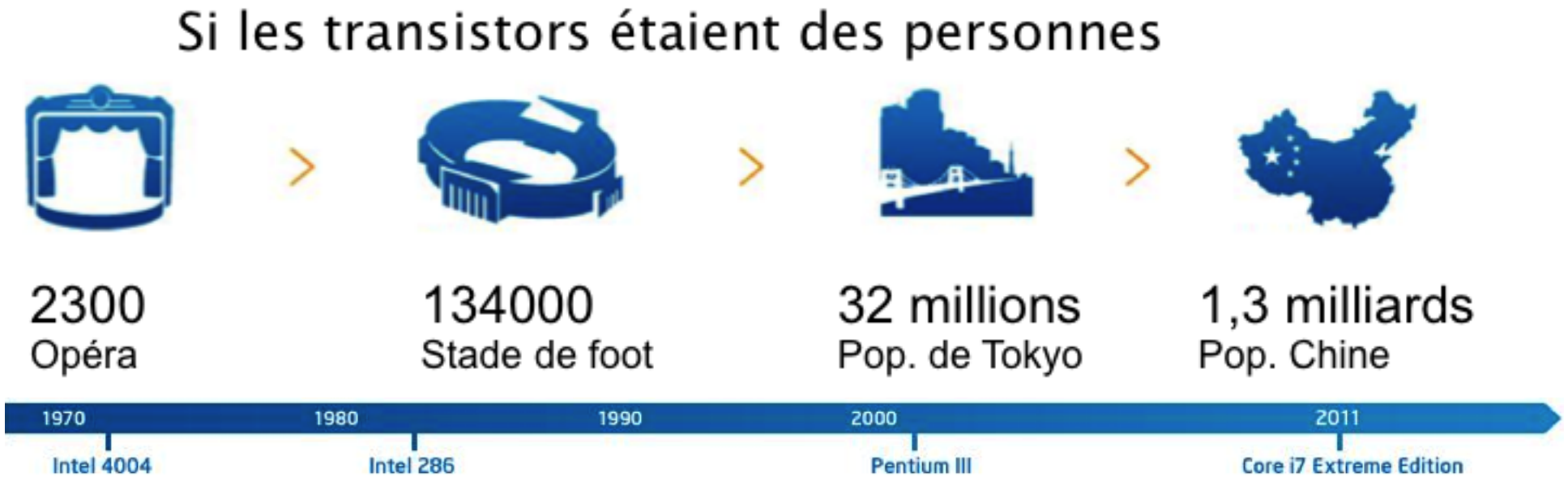
La loi de Moore

- Tous les 2 ans, le nombre de transistors que l'on peut mettre dans un processeur double à prix constant (depuis 1965)



(c) ISSCC 2017

La loi de Moore



Toujours sur la surface d'un opéra...

Loi de Moore

- **Gravure toujours plus fine**: limite 0.2nm
- **Plus d'augmentation de fréquence:** fin de gains faciles !
- Consommation d'énergie devient **critique**
  - Limite augmentation de fréquence pour production de masse de portables, PCs
  - Itanium a atteint 120W, Power6 150W, Power 7 800W (4 chips)
- Coûts de fabrication double tous les 2 ans: les nouveaux processeurs **demandent des productions importantes** pour amortir les coûts de fabrication
- Augmentation du nombre de transistors va continuer

Que faire de ces transistors ?

Un processeur puissant :

- Capable de réaliser plusieurs instructions à la fois
- Des instructions complexes

Pour aller plus vite :

- Augmenter la fréquence

Que faire de ces transistors ?

Un processeur puissant :

- Capable de réaliser plusieurs instructions à la fois
- Des instructions complexes

Pour aller plus vite :

- Augmenter la fréquence

Exemple : Intel P4, Itanium, DecAlpha

Que faire de ces transistors ?

Un processeur puissant :

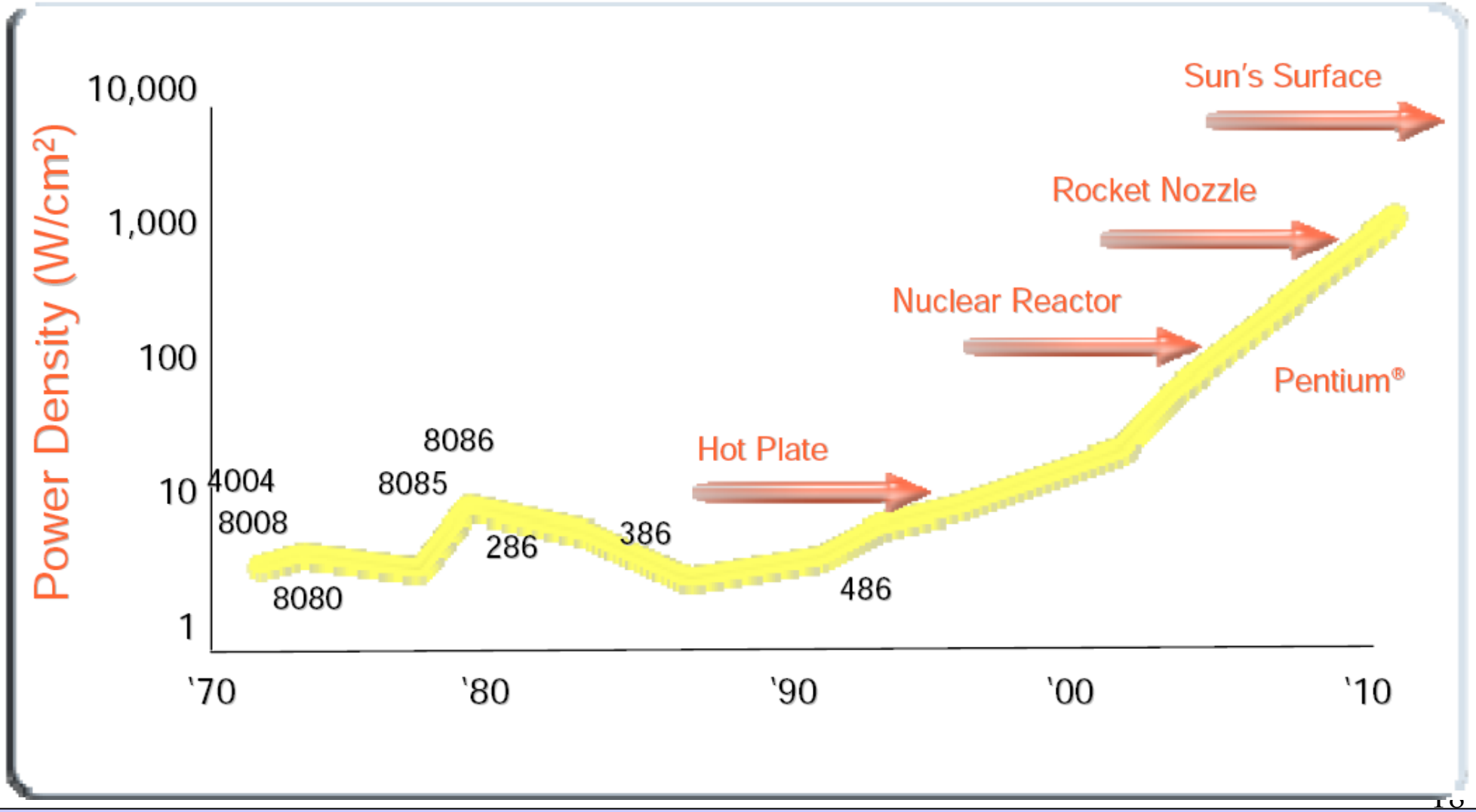
- Capable de réaliser plusieurs instructions à la fois
- Des instructions complexes

Pour aller plus vite :

- Augmenter la fréquence

Problème : Dégage trop de chaleur !

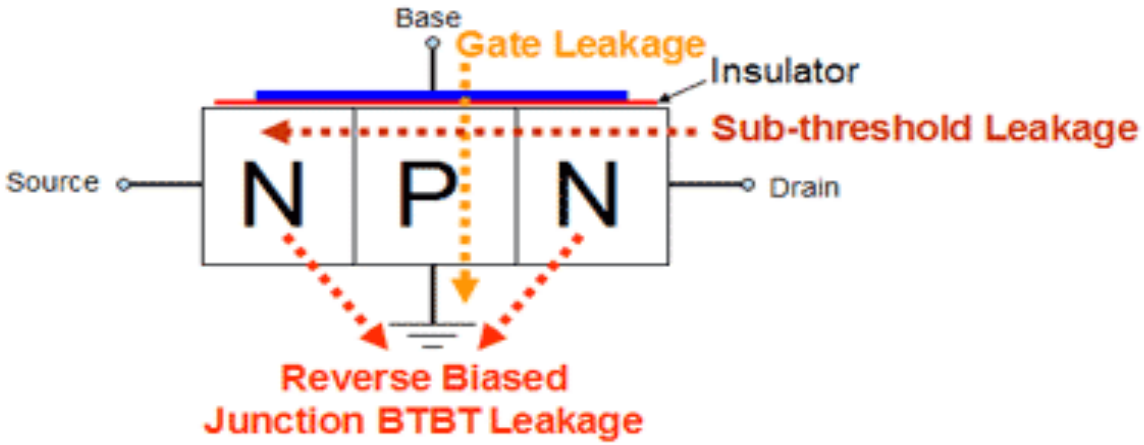
Densité d'énergie libérée en W/cm²



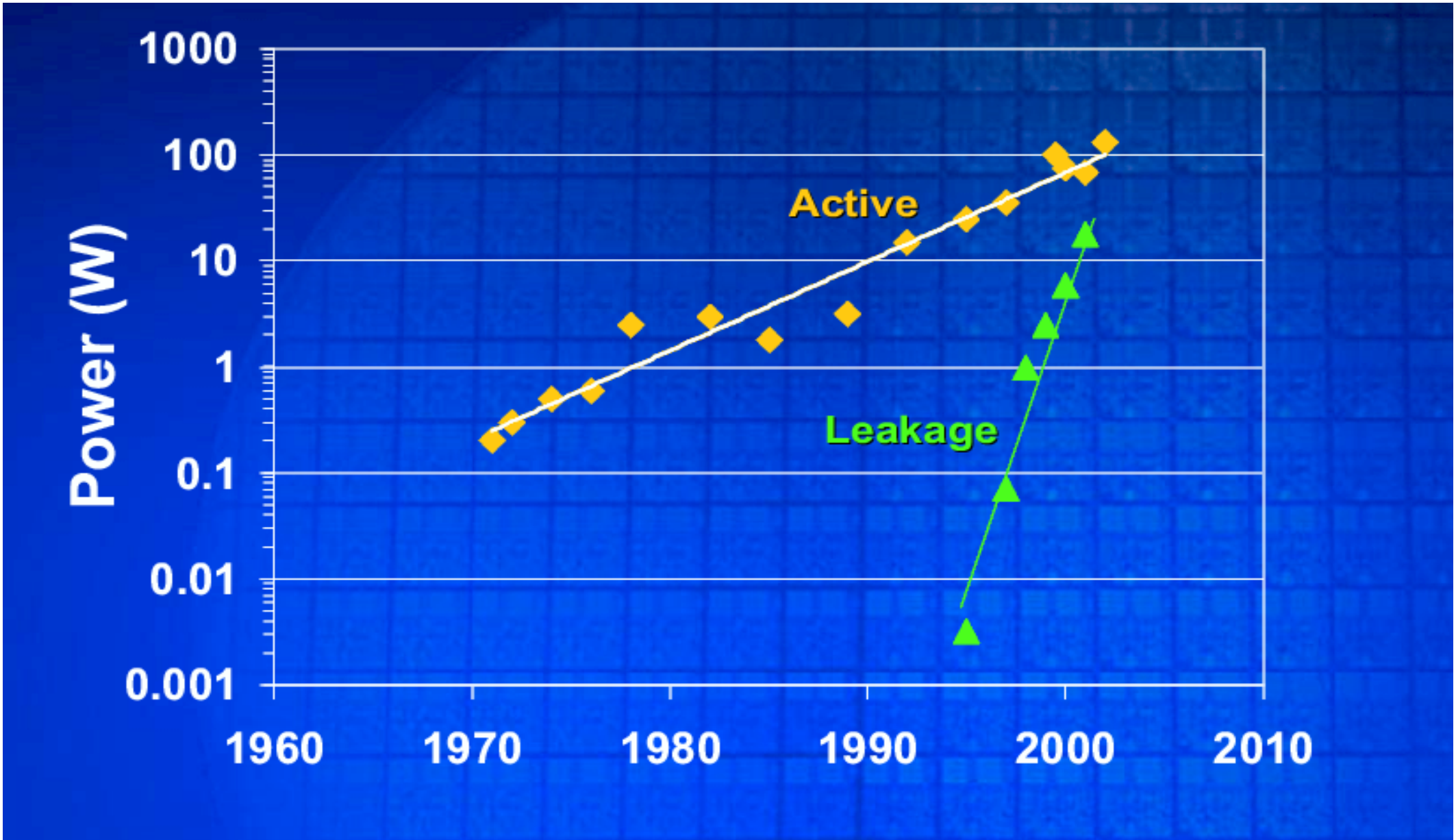
Explication de la puissance dissipée

- Plus le transistor est petit, plus il fuit, par effet tunnel
- Puissance dissipée:

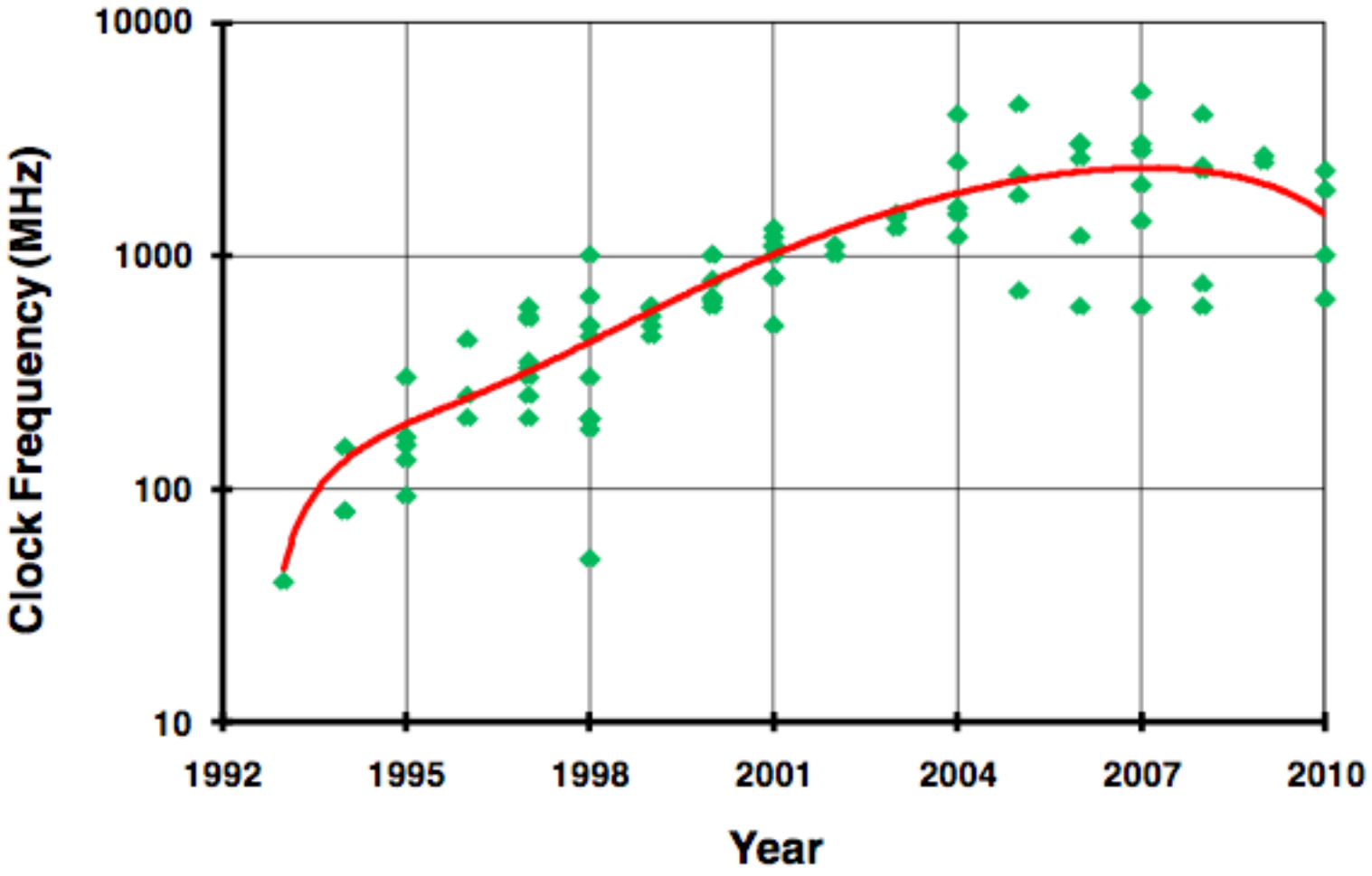
power = $C.f.v^2 + g1 .f. v^3 + g2 .f. v^5$



Evolution de la puissance dissipée



Evolution des fréquences



(c) ISSCC 2017

Que faire de ces transistors ?

Aller plus vite à fréquence constante:

- Augmenter le nombre de cœurs
- Programmation parallèle

Améliorer efficacité énergétique

- Spécialiser des coeurs

Que faire de ces transistors ?

Aller plus vite à fréquence constante:

- Augmenter le nombre de cœurs
- Programmation parallèle

Améliorer efficacité énergétique

- Spécialiser des coeurs

Que faire de ces transistors ?

Aller plus vite à fréquence constante:

- Augmenter le nombre de cœurs
- Programmation parallèle

Améliorer efficacité énergétique

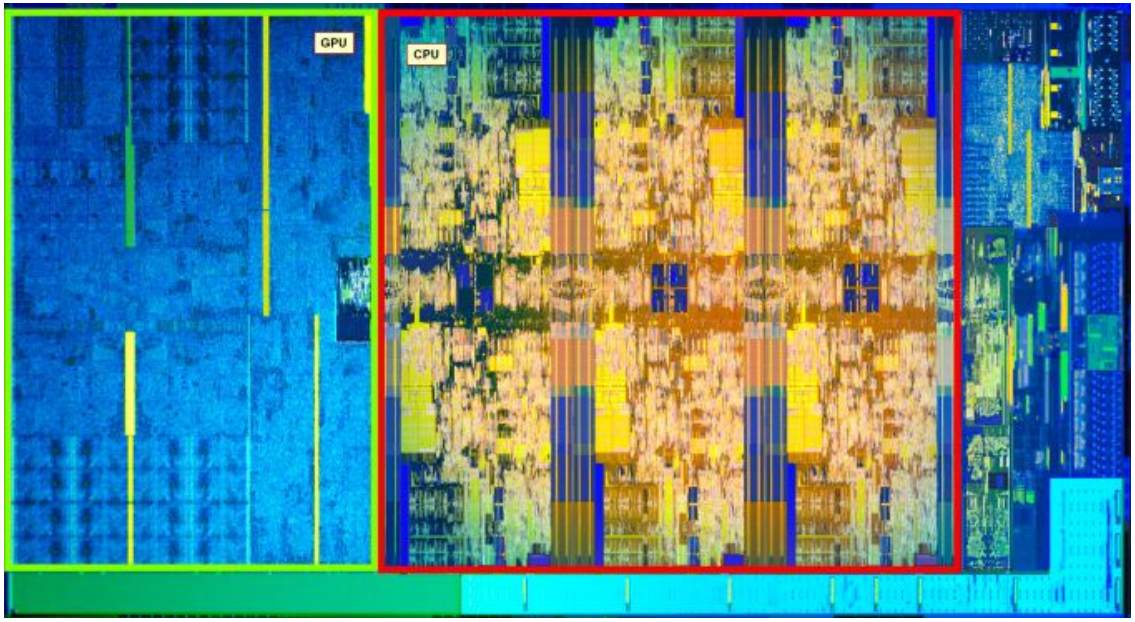
- Spécialiser des coeurs

Passage à l'ère des multicoeurs

Parallélisme multi-coeur partout

Iphone:
6 coeurs ARM

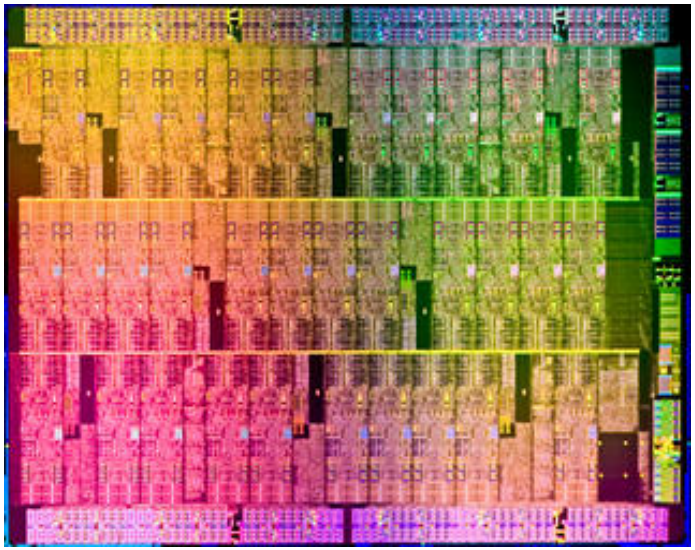
Intel Coffee Lake (6 coeurs, 1 GPU, 95W)



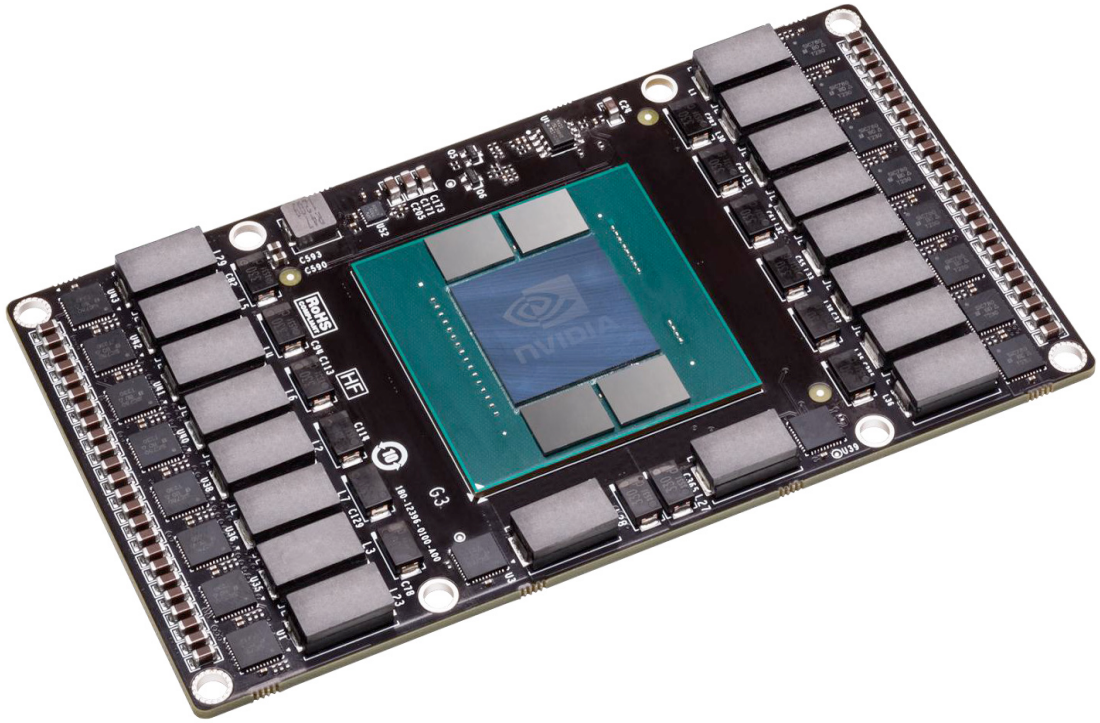
Sony PS4:
8 coeurs AMD

Parallélisme multi-coeur partout

Nvidia Pascal:
64 coeurs
15,3 milliards de transistors, 300W

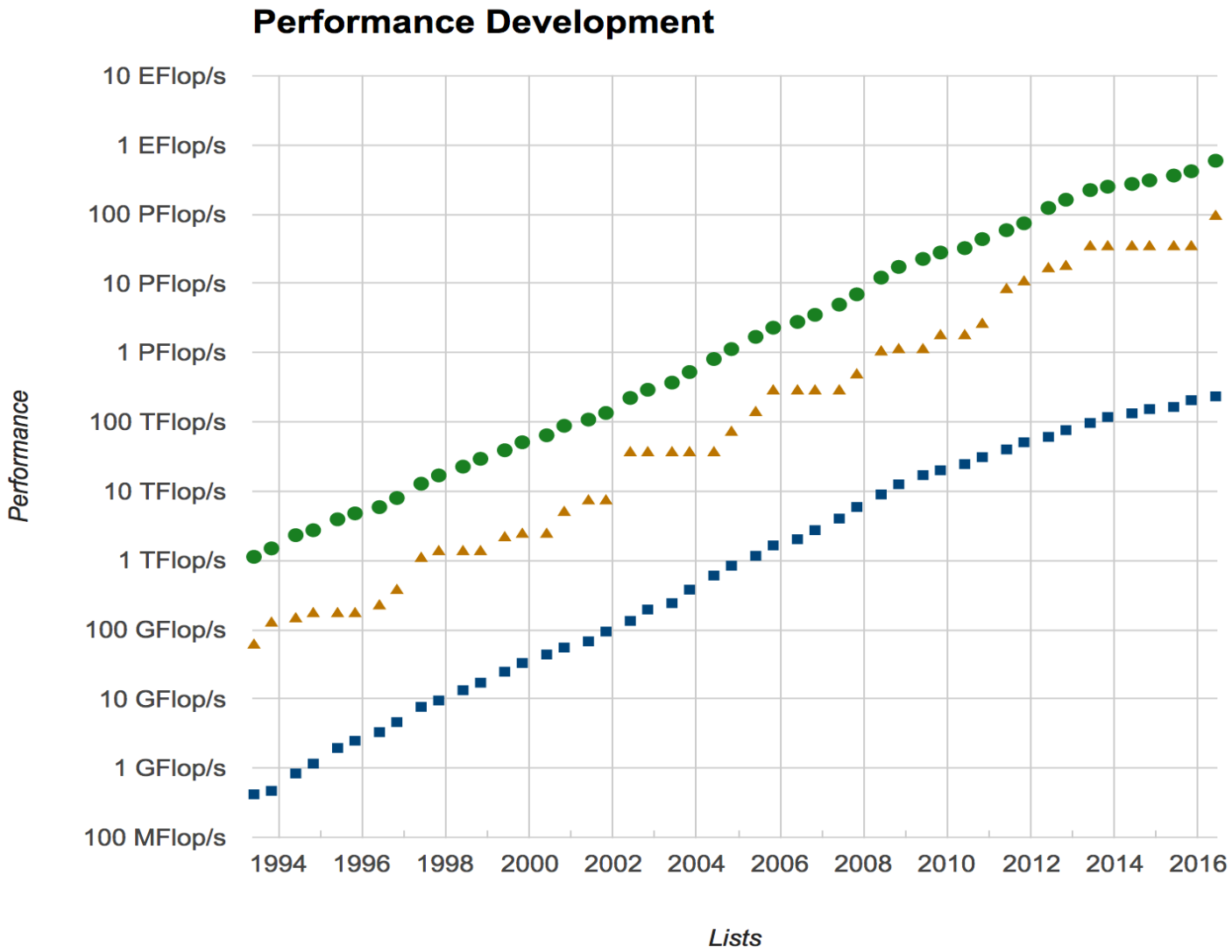


Intel Xeon Phi
72 coeurs, 245W



Tendances sur les performances

Performances pour multicoeurs et machines parallèles sur code scientifique

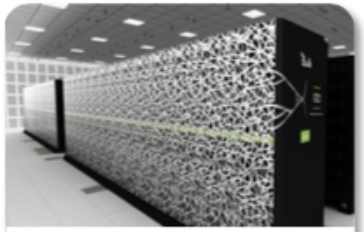


De nouveaux usages

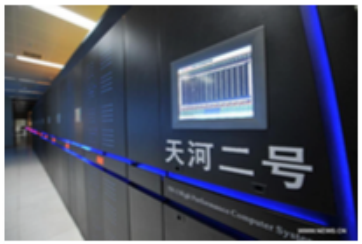
High Performance Computing

1946
ENIAC, vacuum tube computer, 5KOPS, 150KW

1965
General Electric GE635, 4 processors, 2 MIPS



2012
Bull B510, 10K cores, 4TeraBytes RAM, 200 TFlops

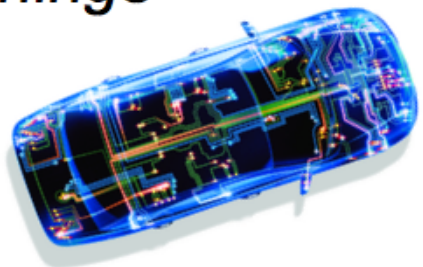


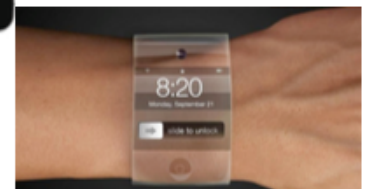
2015
Tianhe-2 (MilkyWay-2): Intel Xeon E5-2692 12C 2.200GHz, Intel Xeon Phi 31S1P, 3.12M cores, 1,024 TB RAM, 50 PFlops, 17,8 MW

Yesterday

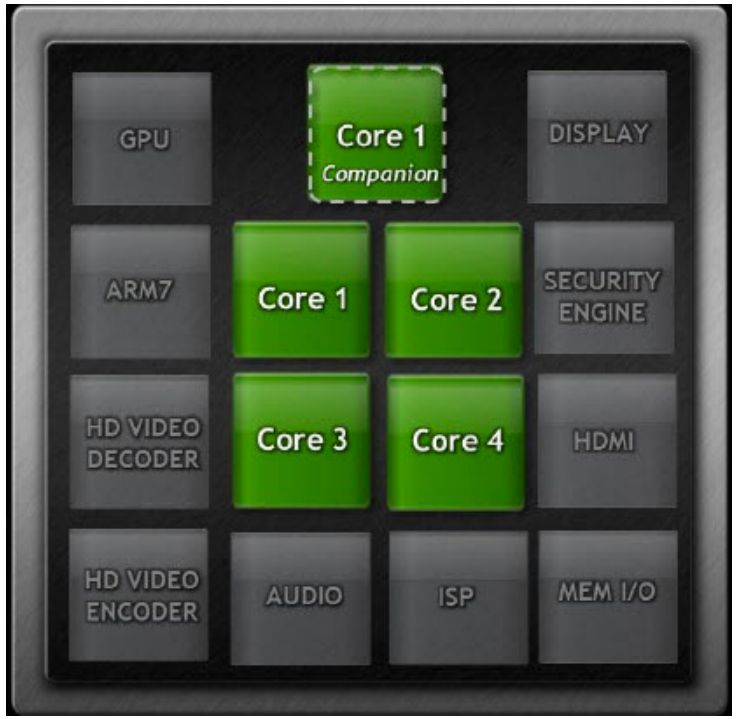
Today

Things



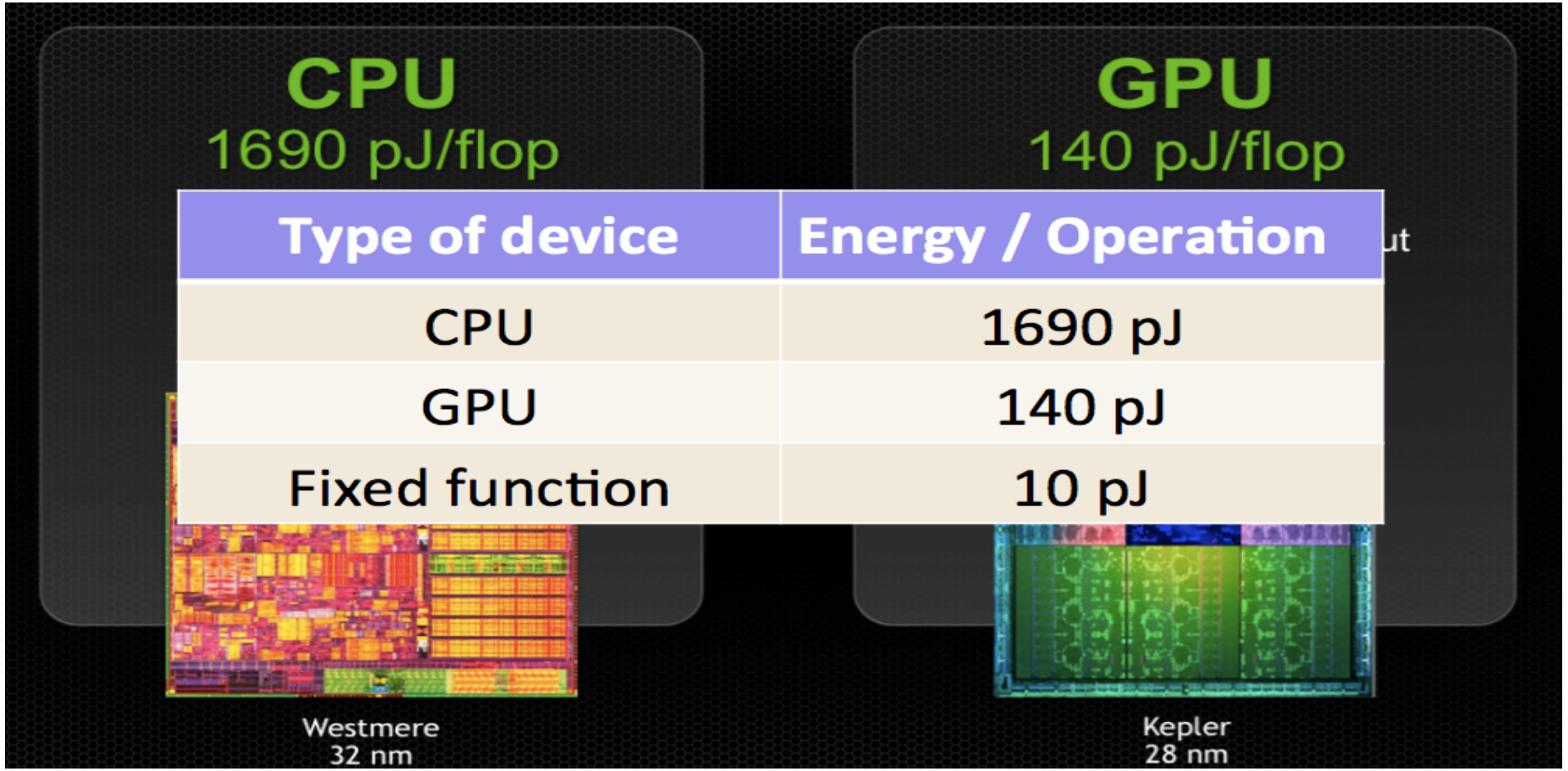


Spécialisation et parallélisme



- Architectures big.LITTLE chez ARM
- Graphic Processing Units
- Architectures hétérogènes (avec plusieurs types de coeurs)

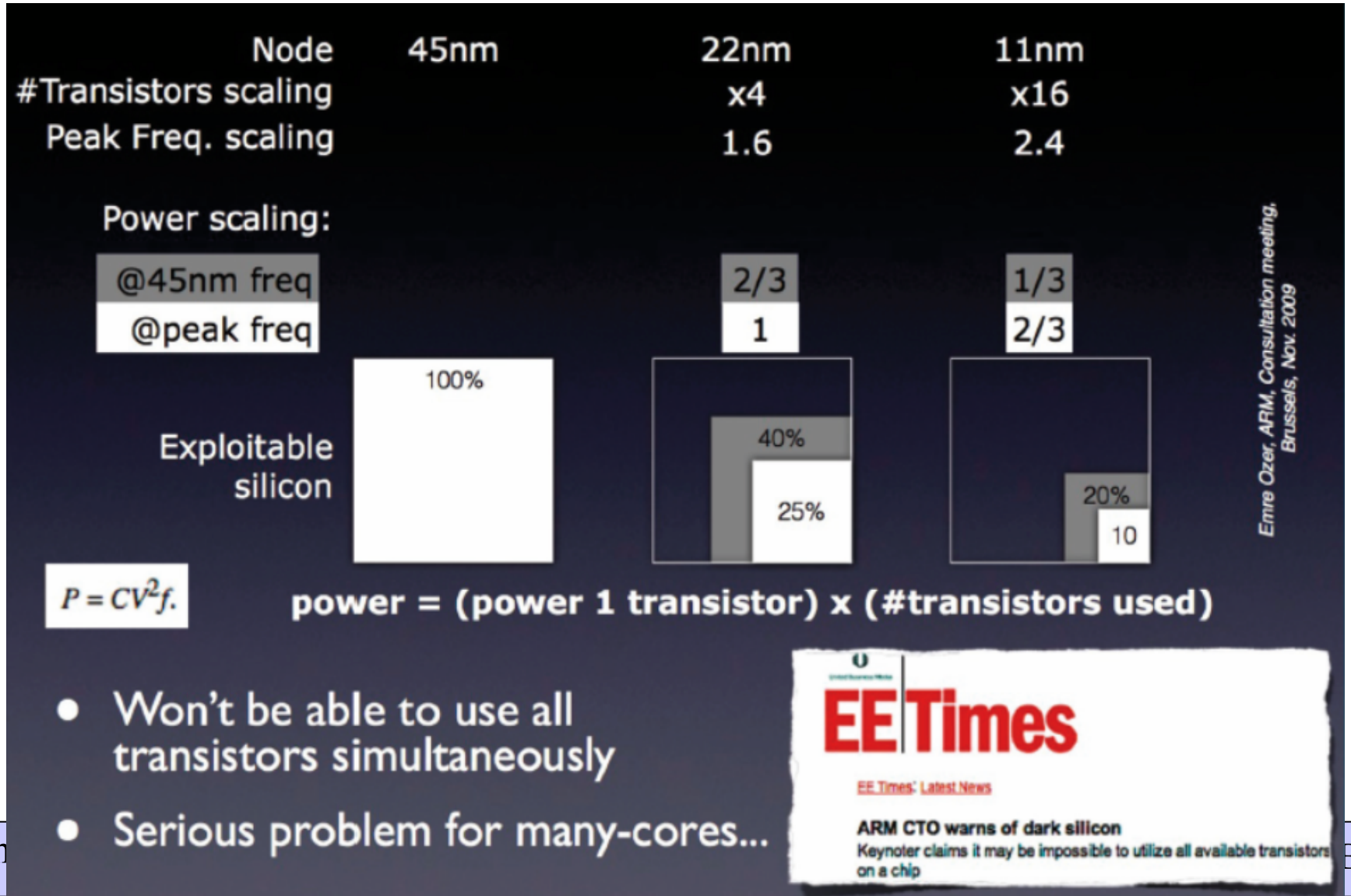
Spécialiser des coeurs pour améliorer l'efficacité



Source from Bill Dally (nVidia) « Challenges for Future Computing Systems »

HiPEAC conference 2015

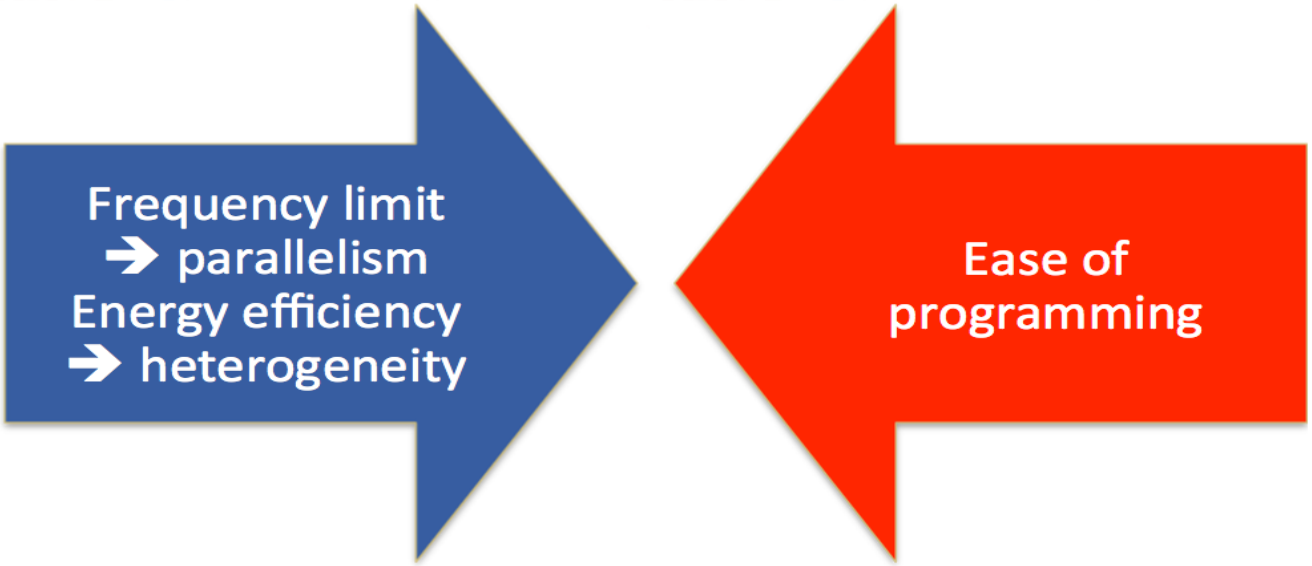
Mieux gérer les coeurs en fonction des besoins



Fin de l'ère monocoeur

- **Rupture très importante**, concerne toutes les couches
  - Impact sur les applications
    - **Pourra t on trouver du parallélisme pour toutes les applications ?**
    - **Performances n'augmentent que si parallélisme augmente**
    - **Changer de façon de programmer**
  - Impact sur les langages/compilateurs
    - Trouver du parallélisme automatiquement
- Davantage de transistors par mm2 pour les années à venir, mais **l'énergie dépensée est un facteur limitant**

Challenges posés par l'évolution des architectures



- Où trouver autant de parallelisme ?
- Revoir les algorithmes, la façon de programmer
- Obstacles culturels, économiques, techniques

Challenges posés par l'évolution des architectures

- **Obstacles techniques:**
  - Les composants architecturaux, mémoire, processeurs, n'évoluent pas de la même façon
  - Changement très rapide des architectures
- **Obstacles culturels:**
  - Maitrise des algorithmes actuels, parallélisme
  - Pratiques de développement, génie logiciel
  - Interdisciplinarité
- **Obstacles économiques:**
  - Difficulté à estimer le retour sur investissement
  - Quelle abstraction et quel développement pour avoir des codes fonctionnant sur plusieurs générations d'architectures ?

Résumé introduction

- La structure des ordinateurs (et donc l'informatique) est durablement marqué par la loi de Moore
- Le parallélisme est désormais le principal facteur d'expension des performances

Loi de N. WIRTH: "Software gets faster slower than hardware gets faster"