« Choix, installation et exploitation d'un calculateur »
Architecture d'aujourd'hui et de demain

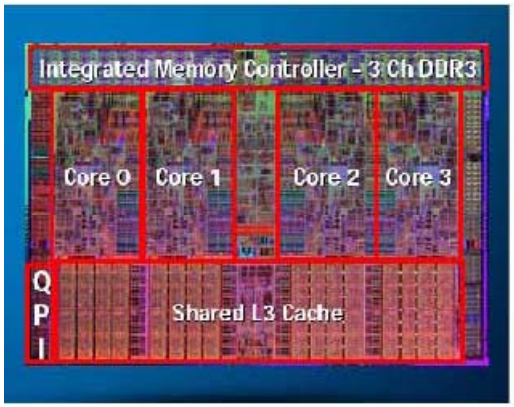
Françoise Berthoud[1]
Violaine Louvet [2]
Françoise Roch [3]

[1]Laboratoire de Physique et de Modélisation des Milieux Condensés -CNRS
[2]Institut Camille Jordan -CNRS
[3]Observatoire des Sciences de l'Univers de Grenoble

Introduction

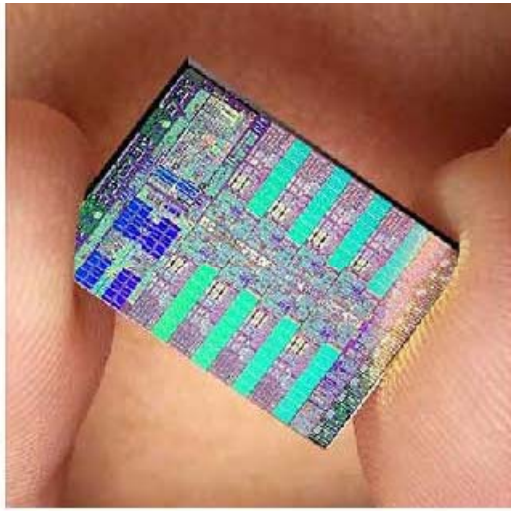
Décoder la relation entre l'architecture et les applications :

- Adapter les algorithmes, la programmation,
- Comprendre le comportement d'un programme,
- Choisir son infrastructure de calcul en fonction de ses besoins, retenir le meilleur compromis



Nehalem


GeForce 8800


Cell

Sommaire

Evolution des architectures

Comment faire des processeurs plus rapides ?

- Augmenter la fréquence d'horloge
- Parallélisme interne des instructions et des données
- Thread Level Parallelism

La problématique de la mémoire

- Les différentes technologies
- Hiérarchisation de la mémoire
- Notion de localité mémoire
- Organisation des caches
- Mémoire virtuelle

Communications CPU -Mémoire -IO

- Bus mémoire
- Bus d'extension

Comment faire des architectures plus rapides ?

- Les architectures multicœurs
- Organisation des caches
- Les processeurs spécialisés

Technologie réseau

Le stockage

Conclusions

Evolution des architectures

- Loi de Moore : nb de transistors par circuit intégré * 2 tous les 2 ans.
- En 10 ans, la finesse de gravure est passée de 0,25 μm (Pentium III) à 0,045 μm (Core 2 Penryn), Nehalem.
- Evolution de l'architecture des processeurs (pipeline, duplication des unités fonctionnelles, exécution spéculative, exécution désordonnée ...).
- Evolution des interactions architecture/applications : dans le cas de l'Itanium, le compilateur fait partie intégrante de l'architecture.
- Evolution des infrastructures vers un haut degré de parallélisme en intégrant un grand nombre de cores.
- Utilisation de cores hétérogènes (CPU + GPU) et API compatibles (CUDA, Brook+, OpenCL)

Evolution des architectures : kilo, méga, giga, téra, péta, …

Top 500 Bench Linpack	Puissance soutenue en Gflops	Puissance crête en Gflops	Nombre de processeurs ou de cœurs
Juin 1993	59.7	131	1024
Juin 2008	1 026 000	1 375 780	122 400

× 1000

1997 : année du Teraflops

× 1000

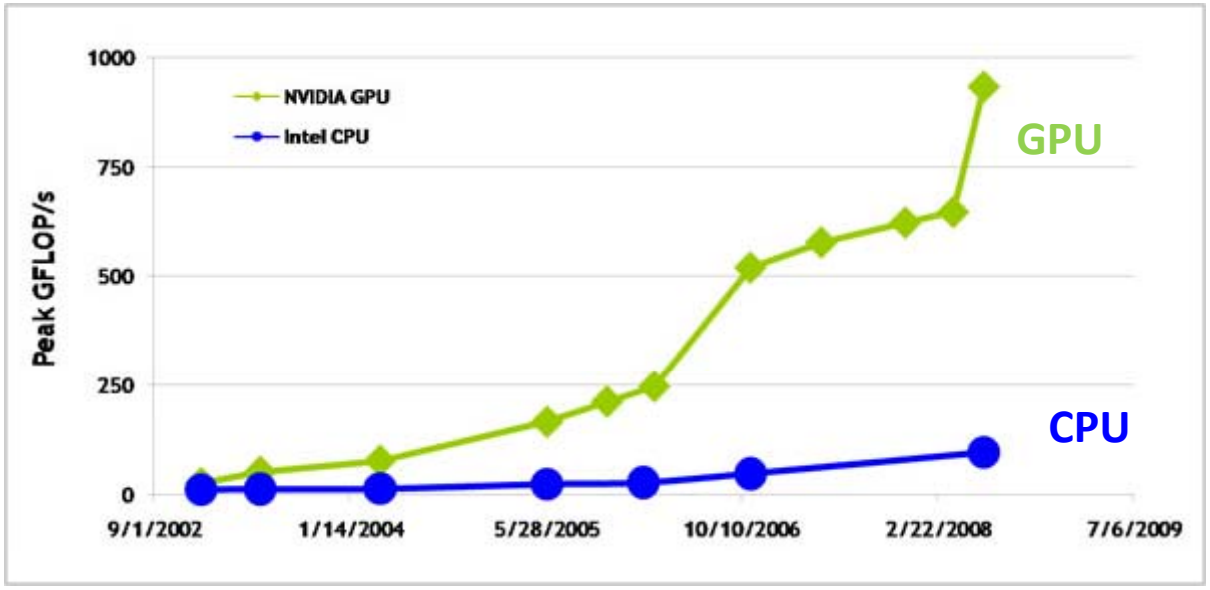
2008 : année du Petaflops

Tests Linpack sur la machine IBM Roadrunner du DoE de Los Alamos

Folding@Home, BOINC : projets de calcul bénévole

2017 : année de l'Exaflops ???

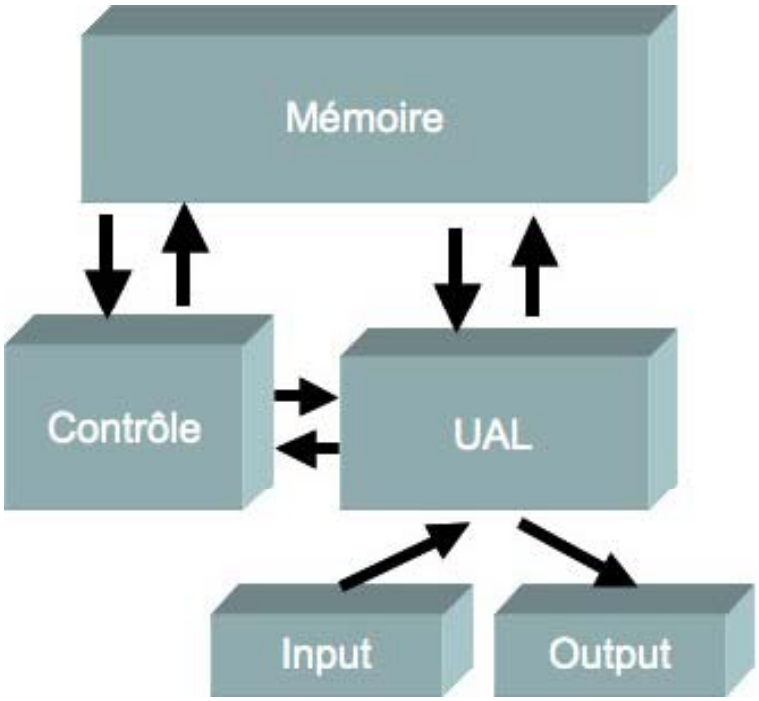
Evolution des processeurs



Vitesse des processeurs classiques *** 2 tous les 16 mois**

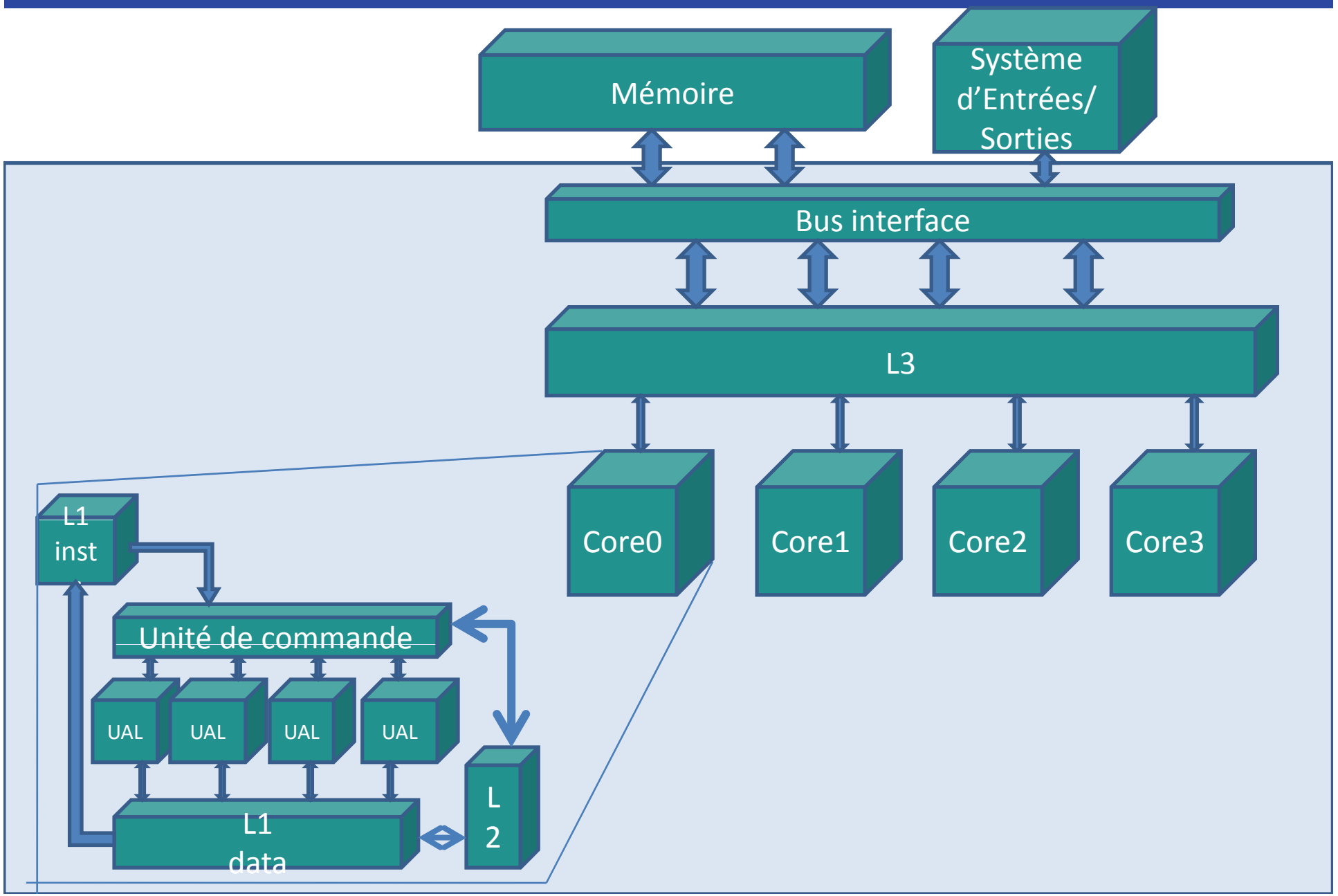
Vitesse des processeurs graphiques (GPU) ***2 tous les 8 mois**

Modèle de Von Neumann (1945)



- La mémoire : contient le programme (instructions) et les données.
- Une Unité Arithmétique et Logique : UAL qui effectue les opérations.
- Une unité de contrôle : chargée du séquençage des opérations.
- Une unité d'Entrée/Sortie.

Une architecture d'aujourd'hui



Classification de Flynn

- Classification des architectures selon 2 dimensions : instruction, data.
- Deux états possibles pour chacune des dimensions : single, multiple.

SISD **Single Instruction, Single Data**	**SIMD** **Single Instruction, Multiple Data**
MISD **Multiple Instruction, Single Data**	**MIMD** **Multiple Instruction, Multiple Data**

Comment faire des processeurs plus rapides

- Augmenter la fréquence d'horloge (limites techniques, solution coûteuse).
- Permettre l'exécution simultanée de plusieurs instructions :
  - Instruction Level Parallelism : pipelining, superscalabilité, architecture VLIW et EPIC.
  - Thread Level Parallelism : multithreading et SMT.
- Améliorer les accès mémoire.

Augmenter la fréquence d'horloge

- La fréquence d'horloge détermine la durée d'un cycle.
- Chaque opération utilise un certain nombre de cycles.
- La fréquence d'horloge est fonction de :
  - la technologie des semi-conducteurs,
  - le packaging,
  - les circuits.

Les limites :

- La consommation électrique et la dissipation thermique d'un processeur sont fonction de sa fréquence d'horloge.
- Le temps d'interconnexion ne suit pas la finesse de gravure.
- Le nb de cycles d'horloge pour interruptions, « cache miss »ou mauvaise prédiction de branchement augmentent.

Exécution en parallèle de plusieurs instructions

1 CPU = plusieurs unités fonctionnelles qui peuvent travailler en parallèle :

- une unité de gestion des bus (unité d'entrées-sorties) en interface avec la mémoire vive du système,
- une unité d'instruction (unité de contrôle)) qui lit les données arrivant, les décode et les envoie à l'unité d'exécution,
- une unité d'exécution qui accomplit les tâches que lui a données l'unité d'instruction, composée notamment de :

  une ou plusieurs unités arithmétiques et logiques (UAL) qui assurent les fonctions basiques de calcul arithmétique et les opérations logiques.

  une ou plusieurs unités de virgule flottante (FPU) qui accomplissent les calculs complexes. L'instruction de calcul de base est la multiplication / addition (FMA pour Floating-point Multiply and Add) en double précision.

La performance crête dépend de la taille des registres et est fonction de la précision (SP,DP). Mesure de performance en terme de Mflops : évalue uniquement les opérations sur les flottants, on peut utiliser les MIPS pour mesurer une performance sur le nombre d'instructions par seconde.

Performance crête du processeur Xeon (quadcore) Harpertown

Il comprend 2 FPU et 1 FMA : 4 opérations flottantes/cycle × 2.5 GHz × 4 = 40 GFlops / proc (en DP)

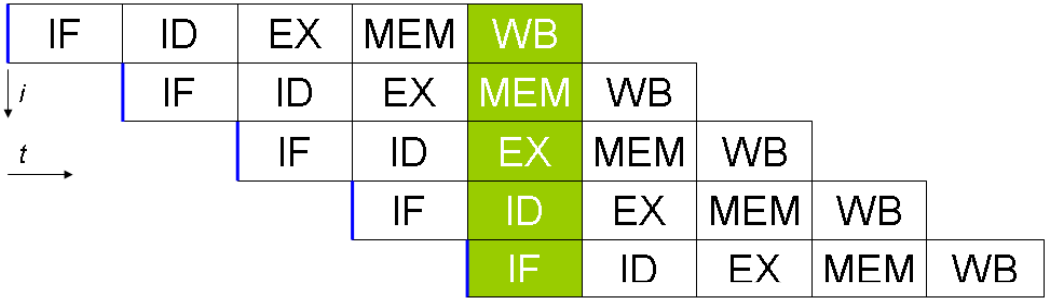
Exécution simultanée de plusieurs instructions : ILP (pipeling)

Principe

Une opération s'exécute en plusieurs étapes indépendantes par des éléments différents du processeur. Les différentes étapes d'une opération s'exécutent simultanément sur des données distinctes (parallélisme d'instructions).

Phases d'exécution d'une instruction :

- IF : Instruction Fecht
- ID : Instruction Decode/Register Fecht
- EX : Execution/Effective Address
- MA : Memory Access/ Cache Access
- WB : Write-Back

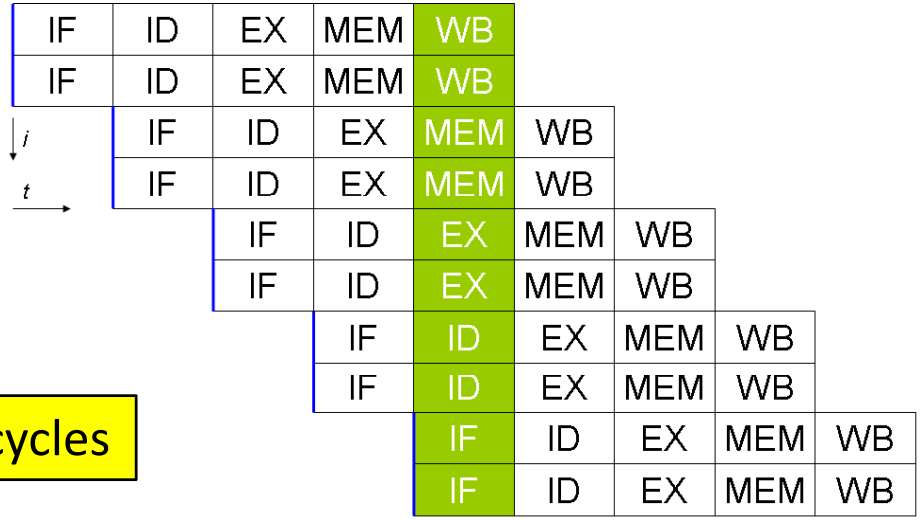


5 instructions en parallèle en 9 cycles (25 cycles en séquentiel) ➔ permet de multiplier le débit avec lequel les instructions sont exécutées par le processeur.

Instruction Level Parallelism : architectures superscalaires

Principe

Duplication de composants (FMA, FPU) et exécution simultanée de plusieurs instructions



10 instructions en 9 cycles

- Gestion des instructions
  - Statique (in-order): exécutées dans l'ordre du code machine
  - Dynamique (out-of-order): le hardware modifie l'ordre des instructions pour favoriser le parallélisme (faible nombre d'instructions)
- Execution spéculative : faire une hypothèse sur la suite d'un traitement après un branchement conditionnel (lorsque la valeur de la condition n'est pas encore calculée).

Les problèmes de dépendances entre instructions :

- Partage des ressources : par exemple la mémoire
- Dépendances des données entre instructions : une instruction produit une opérande utilisée immédiatement par l'instruction suivante
- Dépendances des contrôles : une instruction est une branche.

Appliquer la même instruction à plusieurs données : mode SIMD

- Le mode SIMD (Single Instruction Multiple Data) permet d'appliquer la même instruction simultanément à plusieurs données (stockées dans un registre) pour produire plusieurs résultats.

Exemples de jeux d'instructions SIMD

MMX : permettent d'accélérer certaines opérations répétitives dans des domaines tels que le traitement de l'image 2D, du son et des communications.
SSE (SSE2,SSE3, SSE4 en 2008) : par ex instructions pour additionner et multiplier plusieurs valeurs stockées dans un seul registre (registre SSE)
3DNow : jeu d'instructions multimédia développé par AMD

Les architectures vectorielles

- De nombreux problèmes intrinsèquement vectoriels: ils opèrent sur des vecteurs de données unidimensionnels
- Certaines architectures disposent d'instructions vectorielles
  - l'instruction vectorielle est décodée une seule fois, les éléments du vecteur sont ensuite soumis un à un à l'unité de traitement
  - Les pipelines des unités de traitement sont pleinement alimentés
- Exemple : Cray, NEC SX, Fujitsu VPP, Altivec (PowerPC G4 et G5, Power6)

La tendance est au renforcement des instructions multimédia, à l'augmentation de la taille et du nombre de registres vectoriels

Les limites du parallélisme d'instructions

Obstacles à la performance sur les systèmes superscalaires «standard»:

- Les branchements (IF, CASE, …)
  - Les mauvaises prédictions : en cas d'erreur de prédiction, il faut revenir en arrière
  - Les petites branches ont peu de code à exécuter ce qui limite le parallélisme
- La latence de la mémoire
  - Utilise plus d'un cycle d'horloge
- L'extraction du parallélisme des instructions
  - Le compilateur « sérialise » le code, dont le parallélisme intrinsèque doit être redécouvert dynamiquement par le processeur

Dépasser les limites en « aidant » le processeur

- Les architectures VLIW (Very Long Instruction Word) et EPIC (Explicitly Parallel Instruction Set Computing) permettent de traiter des instructions longues, aggrégations d'instructions courtes indépendantes, de façon à les exécuter explicitement en parallèle.

VLIW : le compilateur a la charge d'organiser correctement les instructions parmi les bundles, tout en respectant les types de dépendances habituelles qui sont normalement gérées au niveau matériel sur les architectures classiques.

→ On gagne en performance (pas de contrôle), mais la compatibilité binaire en générations successives est quasi-impossible.

Dépasser les limites en « aidant » le processeur

- EPIC (Explicitly Parallel Instruction Set Computing) : type d'architecture de microprocesseurs utilisée notamment dans les Itaniums. Le parallélisme est exprimé de manière indépendante de la mise en oeuvre du processeur. Disparition du réordonnancement à l'exécution : les instructions sont exécutées dans l'ordre exact dans lequel le compilateur les a mis.

  → L'effort d'optimisation repose sur le compilateur qui a la charge d'organiser statiquement les dépendances inter-instructions

Exemple : Sur un Itanium, le flot d'instruction est décomposé en blocs (ou bundles, de taille 128 bits) de 3 instructions successives. Chaque bundle contient également un template spécifiant les dépendances au sein du bloc ou entre blocs successifs. Les blocs dépendants les uns des autres forment des groupes. Les instructions appartenant au même groupe peuvent être exécutées en parallèle en fonction du nombre d'unités fonctionnelles disponibles.

Améliorer le remplissage du flot d'instructions du processeur : TLP (Thread Level Parallelism)

- Idée : mixer deux flux d'instructions arrivant au processeur pour optimiser l'utilisation simultanée de toutes les ressources (remplir les cycles perdus - cache miss, load ...).

Le SMT (Simultaneous Multithreading) partage du pipeline du processeur entre plusieurs threads(d'un même programme ou de deux programmes différents). Les registres et les caches sont aussi partagés.

L'hyperthreading = SMT d'Intel.

Le multithreading dégrade les performances individuelles des threads mais améliore les performances de l'ensemble.

Les différentes technologies mémoire

- Principalement deux types à base de semi-conducteur :
  DRAM (Dynamic Random Access Memory) chaque bit est représenté par une charge électrique qui doit être rafraîchie à chaque lecture/écriture.
  SRAM (Static Random Access Memory) retient ses données aussi longtemps qu'il y a du courant.
- Temps d'un cycle SRAM de 8 à 16 fois plus rapide qu'un cycle DRAM.
- Coût de la mémoire SRAM de 8 à 16 fois plus élevé que la mémoire DRAM.

Solution la plus courante:

1. De 1 à 3 niveaux de SRAM en mémoire cache.
2. La mémoire principale en DRAM.
3. La mémoire virtuelle sur des disques.

Type	Taille	Vitesse	Coût/bit
Registre	< 1 KB	< 1 ns	$$$$
SRAM On-chip	8 KB - 6MB	< 10 ns	$$$
SRAM Off-chip	1 MB - 16 MB	< 20 ns	$$
DRAM	64 MB - 1 TB	< 100 ns	$
Flash	64 MB - 32 GB	< 100 s	0
Disk	40 GB - 1 PB	< 20 ms	~ 0

Hiérarchisation de la mémoire

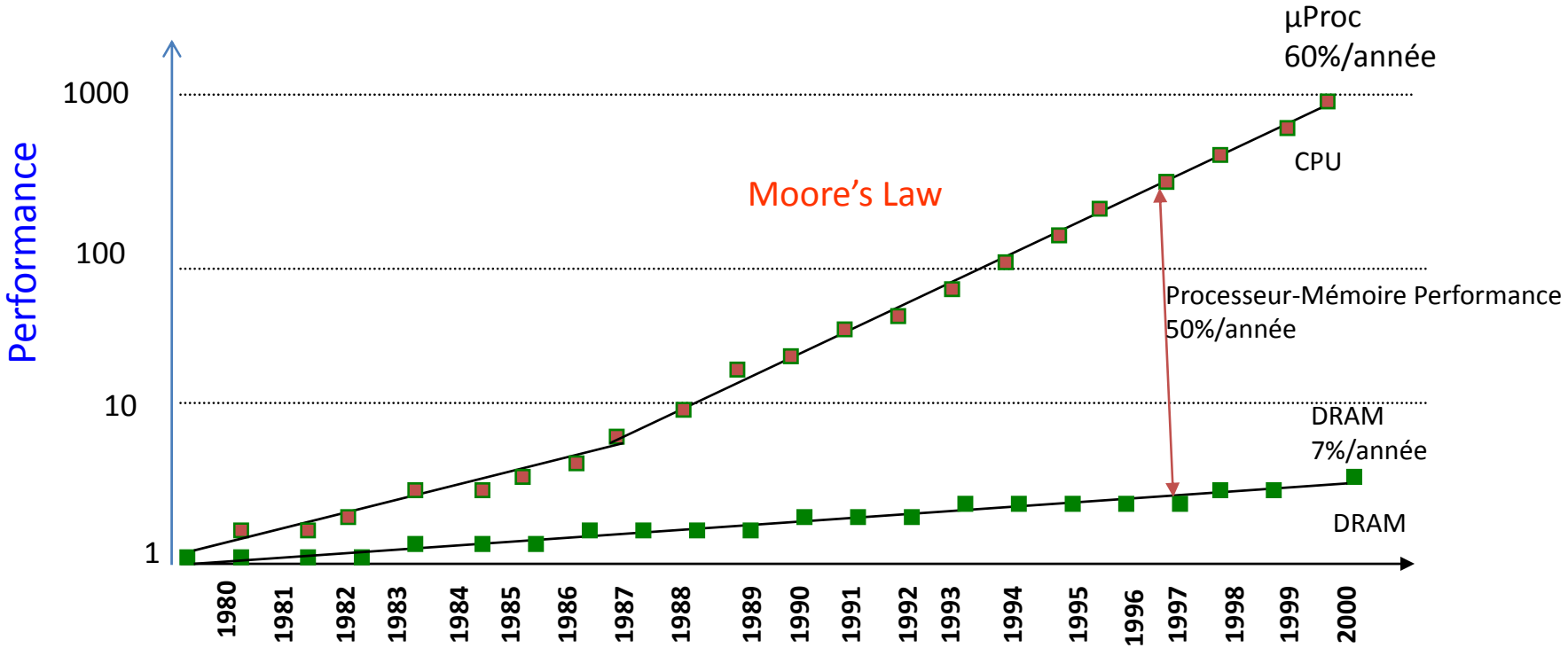
Cas idéal : Mémoire la plus grande et la plus rapide possible

La performance des ordinateurs est limitée par la latence et la bande passante de la mémoire :

- Latence = temps pour un seul accès.

Temps d'accès mémoire >> temps cycle processeur.

- Bande passante = nombre d'accès par unité de temps.



Hiérarchisation de la mémoire

* les caches peuvent être organisés de façon hiérarchique

Notion de localité mémoire : 2 propriétés importantes des références mémoire

- Localité temporelle : si une zone est référencée, elle a des chances d'être référencée à nouveau dans un futur proche.
  - Exemple de proximité temporelle : dans une boucle simple, chaque itération accède aux mêmes instructions.
- Localité spatiale : si une zone est référencée, les zones voisines ont des chances d'être référencées dans un futur proche.
  - Exemple de proximité spatiale : dans un bloc simple, chaque instruction sera accédée l'une après l'autre.

Proximité temporelle

```
DO I = 1,10 0000
      S1 = A(I)
      S2 = A(I+K) # S2 sera réutilisé à l'itération I+K
END DO
```

- L'idée de base est de conserver A(I+K) (lecture de la référence à l'itération I, déclaration S2) en mémoire rapide jusqu'à sa prochaine utilisation à l'itération I+K, déclaration S1.
- Si on veut exploiter la localité temporelle via les registres, cela suppose qu'il faille au moins K registres.
- Cependant, dans le cas général, on aura certainement besoin de stocker d'autres données.

L'évaluation précise de la proximité temporelle est très complexe

Proximité spatiale

- Le voisinage d'une zone localisée par une adresse doit être évaluée dans l'espace d'adressage virtuel.
- Les structures de données sont linéarisées et réduites à une dimension pour être placées dans l'espace d'adressage linéaire.

Exemple en Fortran (rangement des tableaux 2D par colonne)

```
DO J = 1,1000
    DO I = 1,10000
        X1 = A(I,J)
        X2 = B(J,I)
    END DO
END DO
```

- Pour A : localité spatiale parfaite, accès à des zones de mémoire consécutives.
- Pour B : 2 références consécutives sont distantes de 1000 emplacements mémoires.

Pour exploiter la localité spatiale, il faut que la distance entre 2 références soit inférieure à la taille de la ligne du cache.

La hiérarchie mémoire

- Localité temporelle

Garder les éléments récemment référencées aux plus hauts niveaux de la hiérarchie (au plus près du CPU)
=> Les références futures seront rapidement satisfaites

- Localité spatiale

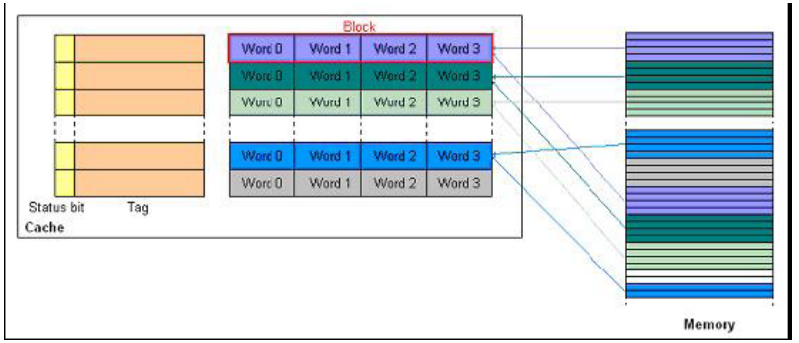
Garder les voisins des éléments récemment référencés au plus hauts niveaux de la hiérarchie
= > les références futures seront rapidement satisfaites

Organisation des caches

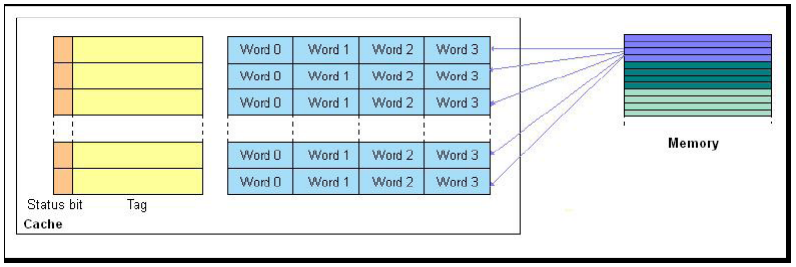
- Le cache est divisé en lignes de n mots.
- 2 niveaux de granularité :
  - le CPU travaille sur des « mots » (par ex 32 ou 64 bits).
  - Les transferts mémoire se font par blocs (par exemple, lignes de cache de 256 octets).
- Une même donnée peut être présente à différents niveaux de la mémoire : problème de cohérence et de propagation des modifications.
- Les lignes de caches sont organisées en ensembles à l'intérieur du cache, la taille de ces ensembles est constante et est appelée le degré d'associativité.

Organisation des caches

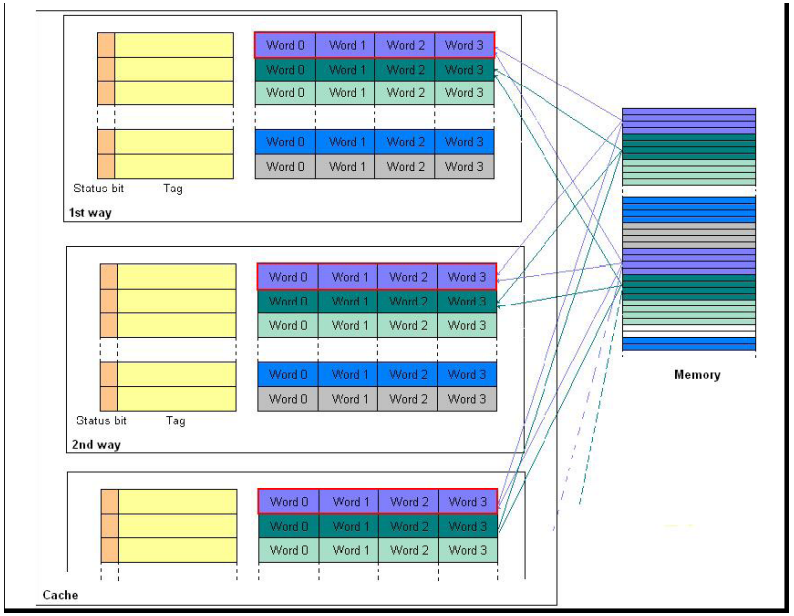
- Direct Map : chaque adresse mémoire est associée à une ligne de cache déterminée (degré d'associativité = 1).



- Fully associative : chaque adresse mémoire correspond à n'importe quelle ligne de cache.



- k way associative : chaque adresse a une alternative de klignes. La mémoire est divisé en ensembles de k lignes.



Lecture/Ecriture des données

Cache hit

Lecture

fournit la donnée à partir du cache

Politique d'écriture :

- **write through : écriture à la fois** dans le cache et la mémoire (simplifie la cohérence) ;
- **write back : écriture seulement** dans le cache, la mémoire est mise à jour une fois la ligne de cache vidée.

Cache miss

Lecture

- récupération de la donnée en mémoire principale
- stockage de cette donnée dans le cache
- fourniture de la donnée à partir du cache

Politique d'écriture :

- **no write allocate : écriture** seulement en mémoire principale ;
- **write allocate : écriture dans le** cache.

Les combinaisons usuelles : « write through » et « no write allocate», « write back » et « write allocate ».

Effets des paramètres des caches sur les performances

- Cache plus grand

☺ réduit les « cache miss »,

☹ augmente le temps d'accès.

- Plus grand degré d'associativité

☺ réduit les conflits de cache,

☹ peut augmenter le temps d'accès.

Prefetching

Le prefetching consiste à anticiper les données et les instructions dont le processeur aura besoin, et ainsi les précharger depuis la hiérarchie mémoire (le L2 ou la mémoire centrale).

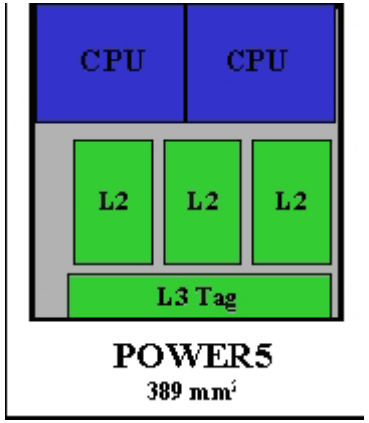
Différents type de prefetching :

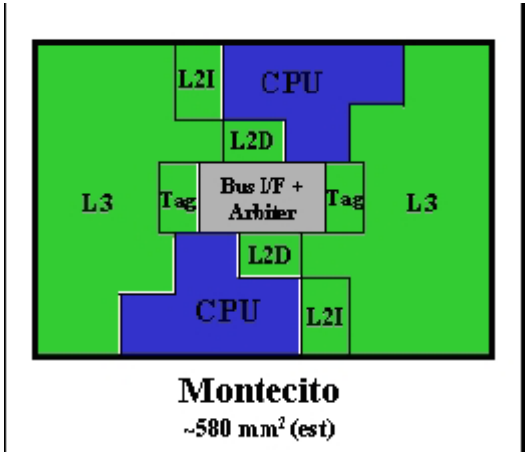
- mécanisme hardware de préchargement,
- mécanisme software de préchargement,
- mécanisme mixte.

Le prefetching n'est pas gratuit, il consomme de la bande passante qui est une ressource critique.

Quelques Exemples

	POWER5	Montecito
L1 I Cache	64 kb, 2 way associative, 4 cycles latency	16 kb, 4 way associative, 1 cycle latency
L1 D Cache	32 kb, 4 way associative, 4 cycles latency	16 kb, 4 way associative, 1 cycle latency
L2 Cache	1.92 MB, 10 way associative, 13 cycles latency	2 1 MB (I), 2 256 kb (D), 8 way associative, 5 cycles latency
L3 Cache	36 MB, 12 way associative, 87 cycles latency	24 MB, 2 way associative, 14 cycles latency
Mémoire	16 GB/s per device, 220 cycles latency	21.3 GB/s per device 224 cycles latency

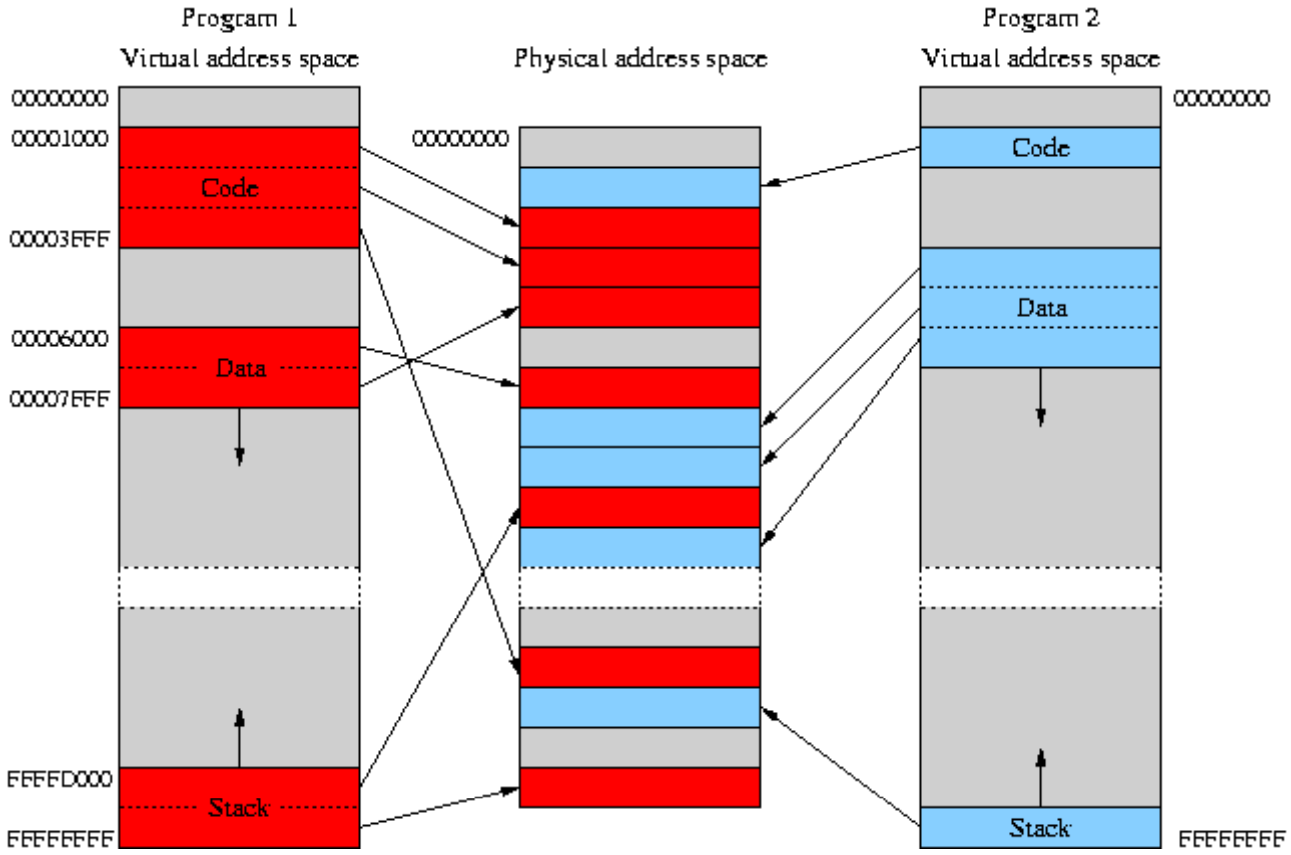




Mémoire Virtuelle

- La mémoire virtuelle est le dernier niveau de la hiérarchie mémoire.
- Espace d'adressage virtuel (ou logique) :
  - Quantité maximale d'espace d'adressage disponible pour une application.
  - Cet espace d'adressage virtuel est en correspondance avec l'espace d'adressage physique pour l'accès aux données via le MMU (Memory Management Unit) au niveau hard, et via le système d'exploitation.
- L'espace d'adressage virtuel de chaque programme et l'espace mémoire sont divisés en pages (physiques ou virtuelles) de même taille.

Correspondance espace d'adressage physique - espace d'adressage virtuel



Pour chaque page virtuelle utilisée par un programme correspond une page physique

Accès aux pages de l'espace physique

- Taille des pages :
  - **X86** pages de 4KB (on peut aussi trouver 2 MB, 4 MB).
  - **NEC SX** supporte à la fois des petites et des grandes pages.
  - **Itanium** plusieurs tailles de page possibles : 8 KB, 16KB, 64 KB, 1 MB, ...
- Le placement est « fully associative ». Une table des pages contient l'ensemble des informations nécessaires à la correspondance adresses virtuelles <-> adresses physiques.
- Problème : 4 GB de mémoire et des page de 4 KB impliquent une table de 1 millions d'entrées !

  →Table de pages multi-niveaux,

  → la table peut être mise en cache.

Page faults

L'adresse virtuelle référence une page qui n'est pas trouvée en mémoire physique :

- Si la mémoire physique est pleine, supprimer une page de la mémoire physique (remplacement) :
  - choisir une page « victime »,
  - si elle a été modifiée, la ré-écrire sur disque,
  - modifier les indicateurs de présence dans la table.
- Dans tous les cas :
  - charger la page référencée en mémoire physique (placement),
  - modifier les indicateurs de présence dans la table.

Les défauts de page sont extrêmement coûteux !

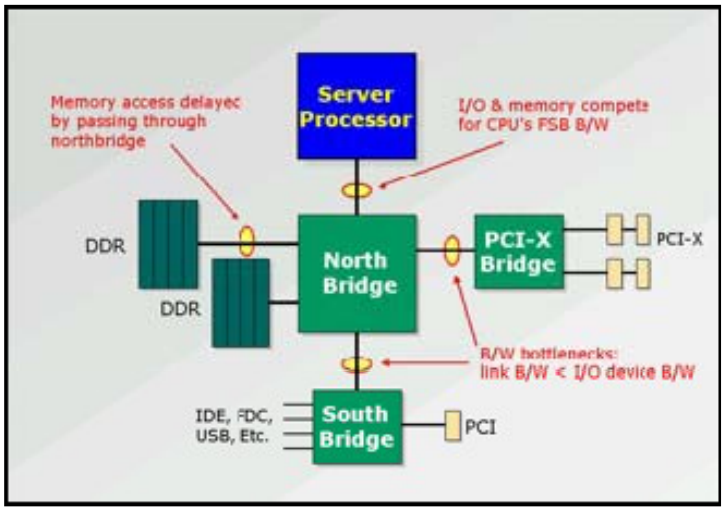
TLB, Translation Lookaside Buffer

- Objectif : Eviter l'accès multiple au cache pour récupérer les @ physiques.
- Idée : utiliser un cache spécifique, le TLB, qui ne référence que les pages virtuelles les plus récentes.

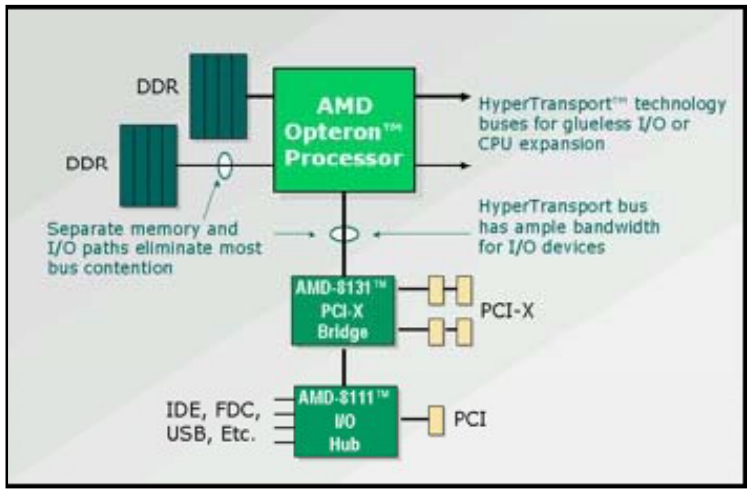
- La taille du TLB est limitée.
- TLB misses : l'adresse n'est pas dans le TLB :
  - le processeur parcourt la table des pages du processus, si la page est dans la table, chargement des informations relatives à cette page dans le TLB. Il faut compter quelques dizaines de cycle pour trouver et charger les infos dans le TLB.
  - Si la page n'est pas en mémoire principale, c'est un défaut de page. Le coût pour récupérer un défaut de page est très élevé.
- Eviter les défauts de TLB :
  - Eviter des accès aléatoires à des ensembles de données de plus de 1 Mo de façon à eviter les "TLB miss".
  - Le TLB est « 2 way-associative »(2 voies de 128), il faut donc éviter des sauts (strides) de taille une grande puissance de 2 pour éviter les TLB miss.

Les communications en interne

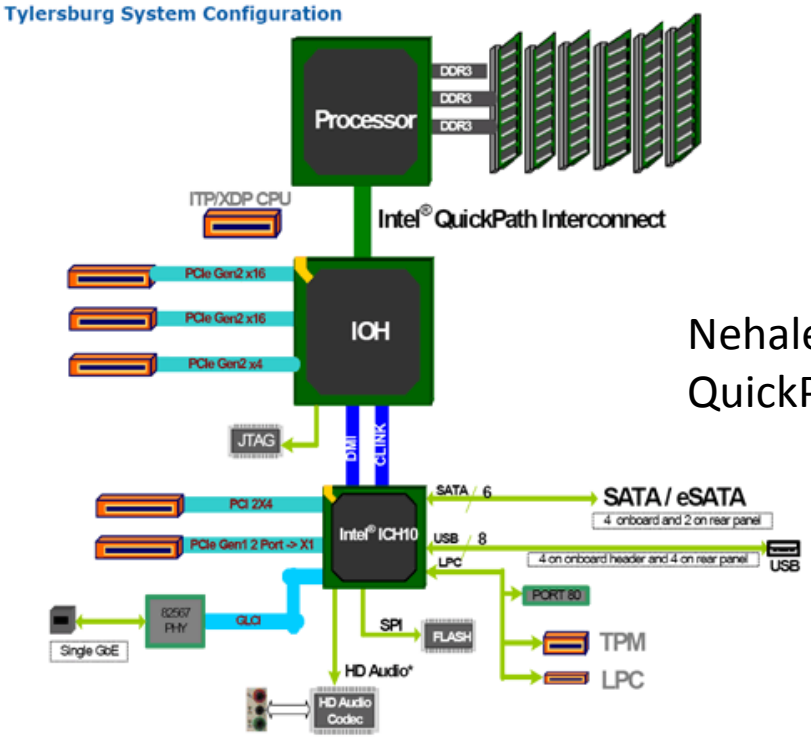
Exemple : xeon versus opteron



Xeon
Northbridge



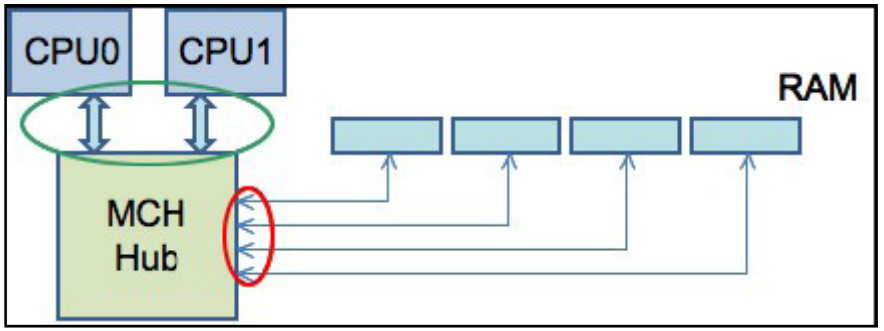
Opteron
Hypertransport



Nehalem
QuickPath

Communications CPU - Mémoire

Exemple du Northbridge sur architecture Intel:
Pour évaluer la bande passante entre CPU et mémoire, il faut
Connaître la fréquence du FSB.
Fréquence la + élevée : 1 600 MHz.



Exemple : calcul de la bande passante mémoire locale sur un noeud bi-sockets Intel Hapertown :
Bande Passante au niveau de l'interface processeurs : sur une carte bi-sockets 2 FSB 64 bits à 1 600 MHz, soit 2*1600*8 = 25.8 Go/s.
Bande Passante au niveau de l'interface mémoire : la mémoire est adressée via 4 canaux à 800 MHz (si FBDIMM 800 MHz). La largeur du bus est de 72 bits, dont 8 bits de contrôle, soit 64 bits de données. Ce qui donne : 4*800*8 = 25.6 Go/s.

Communications CPU - Mémoire - IO

Front Side Bus

Hypertransport (HT), solution AMD liaison point à point

- Connexion haut débit, bidirectionnel, à faible latence.
- Les processeurs sont reliés entre eux via un lien hypertransport à 2 GHz en HT 2.0 (passage à 2.6 GHz en HT 3).

Limite de la bande passante sur hypertransport (V3 en 2.6 GHz) : 20.8 GB/s dans chaque direction soit 41.6 en FD.

Par ailleurs, Chaque processeur dispose de 2 ou 3 canaux de mémoire (contrôleur intégré au processeur).

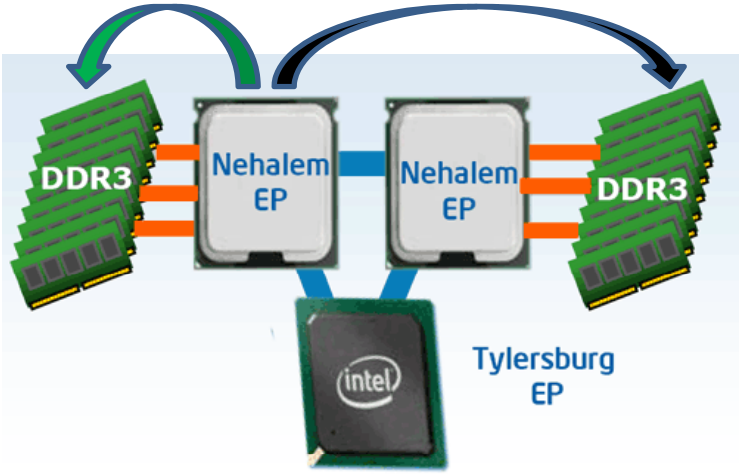
QuickPath Interconnect (QPI), solution Intel

- Remplace le FSB sur l'architecture Nehalem, similaire à l'HT.
- Le principal intérêt du bus QPI provient de sa topologie point à point : le bus connectant les processeurs aux autres composants n'est plus partagé.

Jusqu'à 25.6 GB/s en bi-directionnel.

Communications CPU - Mémoire

Exemple du Nehalem avec le QuickPathInterconnect :
Le contrôleur mémoire est intégré au CPU
.



Exemple : calcul de la bande passante mémoire locale sur un noeud bi-sockets Intel Nehalem

Bande Passante vers mémoire locale au CPU :
3 canaux 64 bits vers mémoire DDR3 1066 MHz, soit 3*1066*8 = 25.6 Go/s.
Un processeur peut également adresser la mémoire connecté à un autre processeur NUMA (Non Uniform Memory Access)
Bande Passante vers mémoire d'un autre CPU : le QPI à une bande passante de 6.4 GT /s par lien, unité de transfert de 16 bits, soit 12.8 Go/s dans chaque sens, soit une bande passante totale de 25.6 Go/s.

Communications CPU – IO

Bus interne d'extension

Bus PCI-X

Evolution du bus PCI, préserve la compatibilité.
Interconnect parallèle. Interface entre le bus processeur et le bus d'entrées/sorties. Bus 64 bits, Fréquence jusqu'à 533 MHz.

Bus série full duplex PCI-Express

Utilise une interface série (1 bit donc) à base de lignes bi-directionnelles. En version 2, chaque lien série fonctionne à 250 Mo/s dans chaque direction. Le bus PCI-Express peut être construit en multipliant le nombre de lignes => débits de 250 Mo/s à 8 Gos/s.

Limite de la bande passante sur PCI-X 133 : 64 bits 133 MHz = 1064 Mo/s

Limite de la bande passante sur PCI-Express (en V2 : 250 Mo/s par canal) : 2X : 0.5 Go/s 4X : 1 Go/s 32X : 8 Go/s

Les architectures multicoeurs

- Un processeur composé d'au moins 2 unités centrales de calcul sur une même puce.
- Permet d'augmenter la puissance de calcul sans augmenter la fréquence d'horloge.
- Et donc réduire la dissipation thermique.
- Et augmenter la densité : les coeurs sont sur le même support, la connectique reliant le processeur à la carte mère ne change pas par rapport à un mono-cœur

La course à la fréquence est terminée !

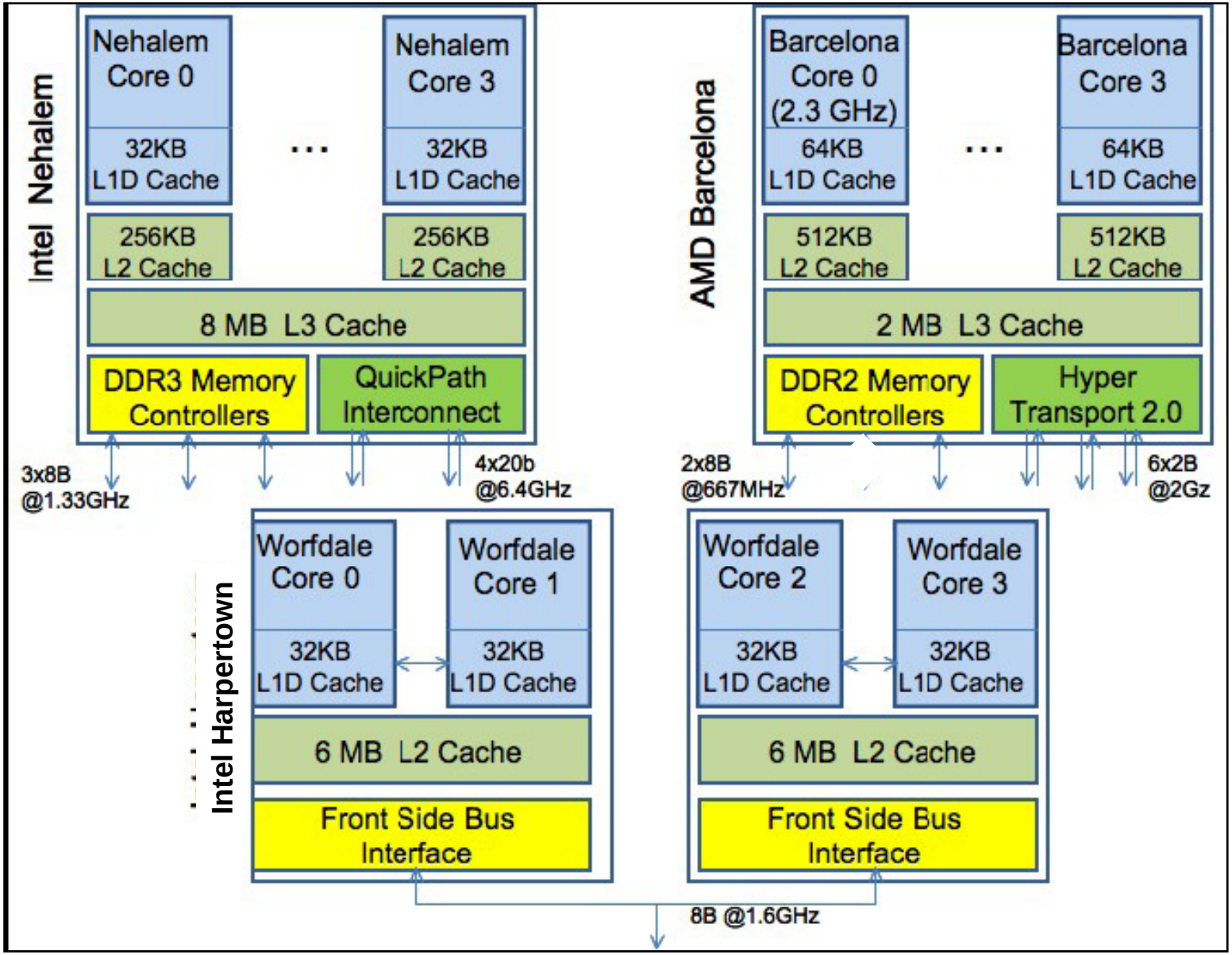
La version originale de la loi de Moore dit que le nombre de transistors des circuits intégrés double tous les deux ans. Que faire de tous ces transistors ?

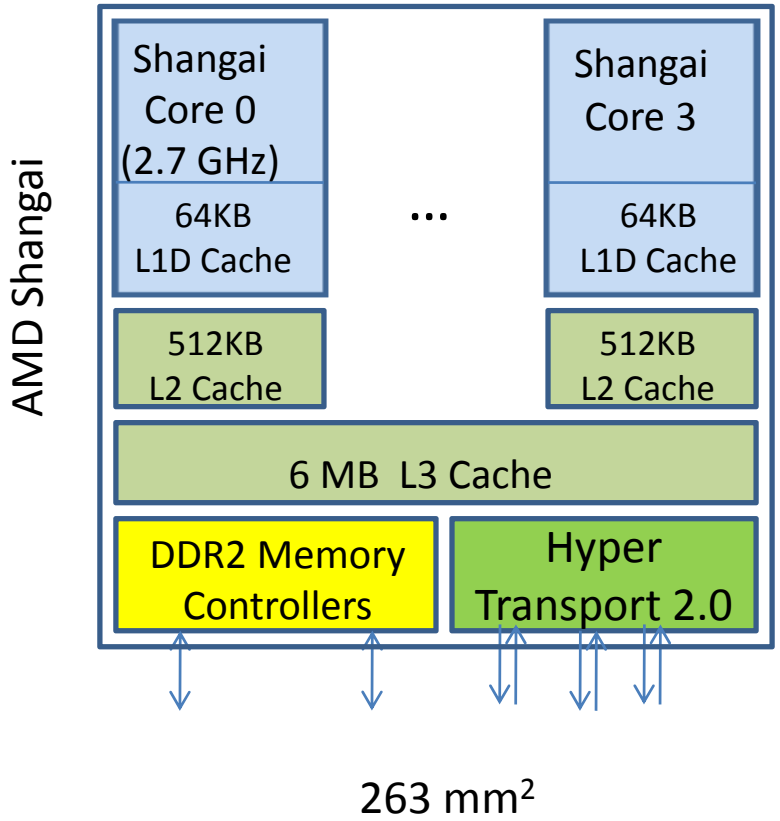
- une complexité sur l'«out of order» accrue mais montre ses limites
- des caches de plus en plus grands (mais qui limitent la vitesse d'accès aux données),
- plus de CPUs.

Exemples

IBM POWER5, POWER6, Intel Core 2 duo, xeon (quad-coeurs Clovertown, Hapertown, Nehalem), AMD opteron double coeurs denmark, quad -coeurs Barcelona,
Intel Itanium Montecito...

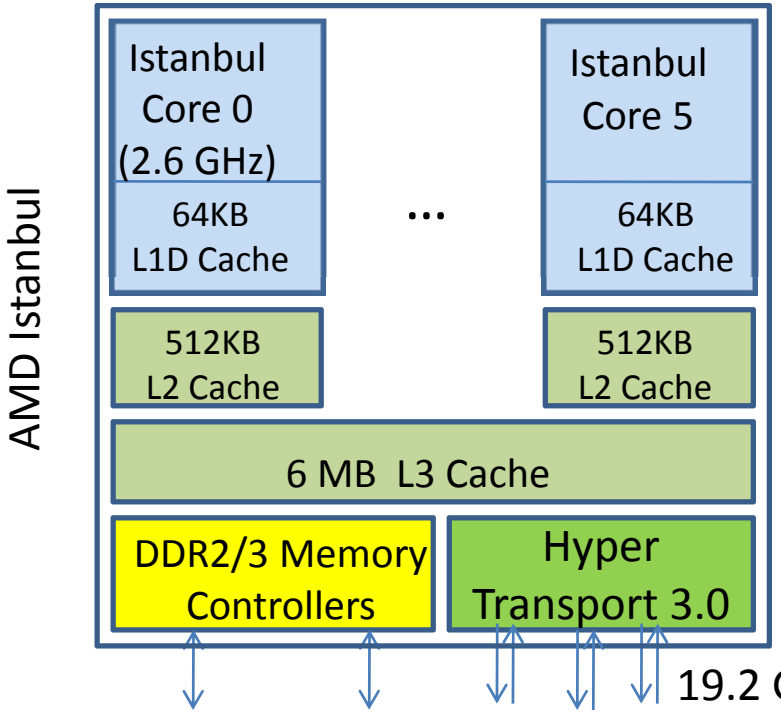
Exemples de processeurs multicoeurs





263 mm²

Seulement 2 canaux DDR2 800 MHz



19.2 GB/s par lien

294 mm²

Processeur Istanbul : 6 cœurs

Pourquoi les multicoeurs ?

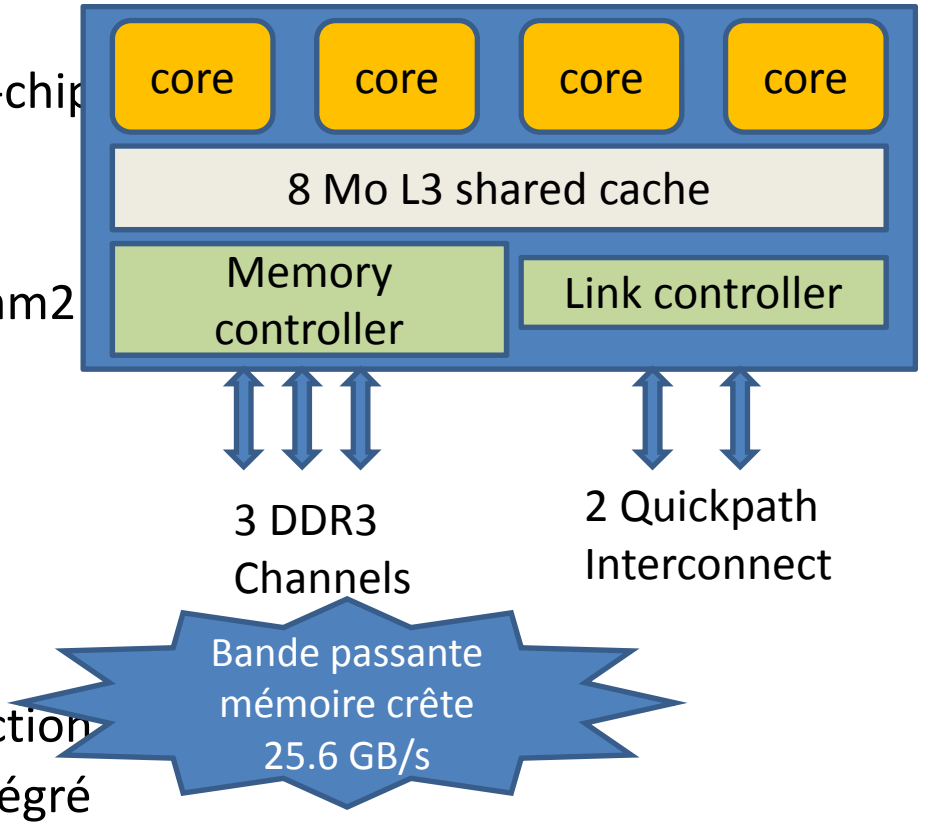
Quelques ordres de grandeur

	Single Core Génération de gravure1	Dual Core Génération de gravure2	Quad Core Génération de gravure3
Core area	A	~ A/2	~ A/4
Core power	W	~ W/2	~ W/4
Chip power	W + O	W + O’	W + O”
Core performance	P	0.9P	0.8P
Chip performance	P	1.8 P	3.2 P

Architecture Nehalem-EP

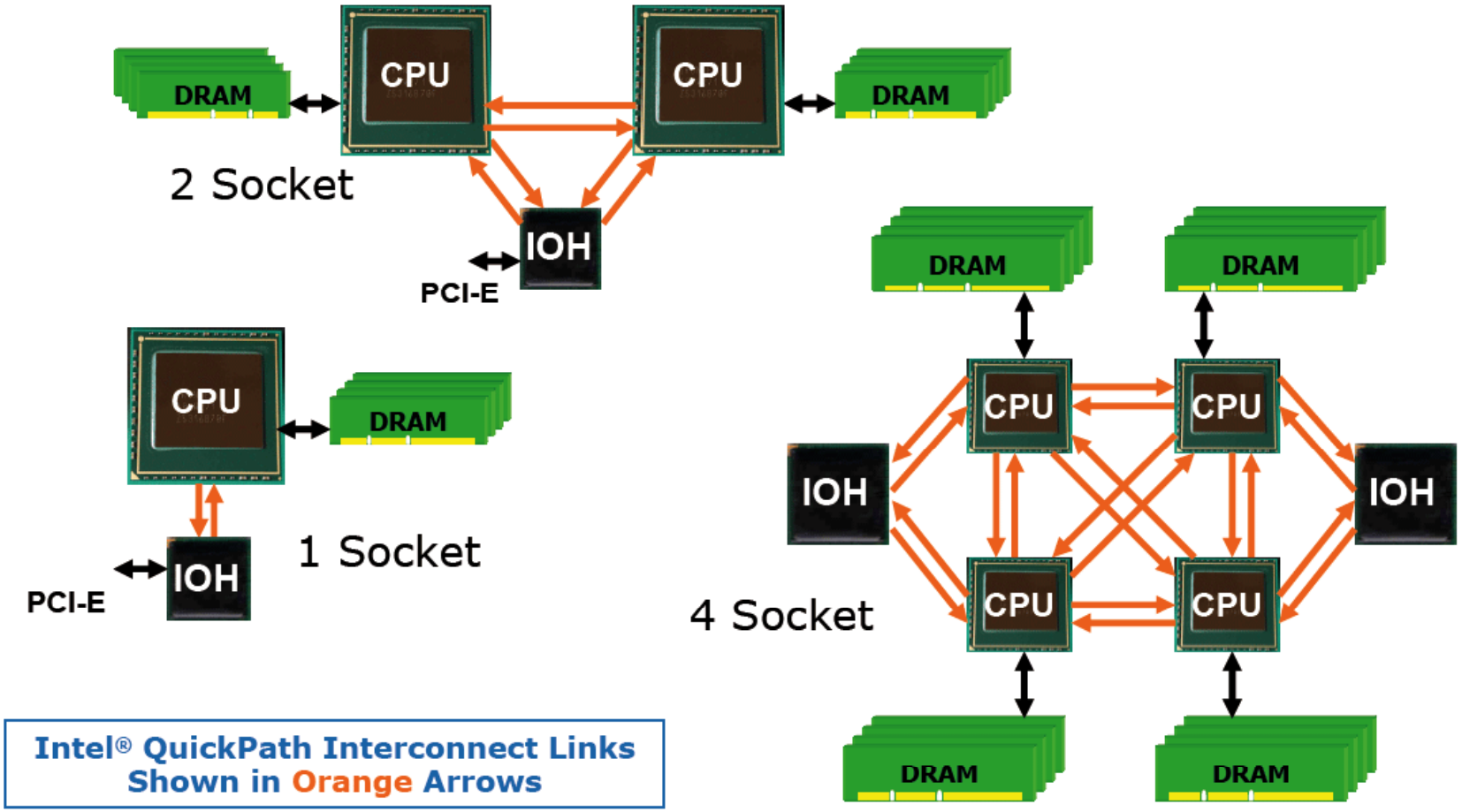
- 4 cores
- Cache L3 partagé 8 Mo on-chip
- 3 niveaux de cache
  Cache L1 : 32k I-cache + 32k D-cache
  Cache L2 : 256 k par core
  Cache inclusif : cohérence de cache on-chip
- SMT

- 732 M transistors, 1 seule puce de 263 mm2
- QuickPathInterconnect
  - Point à Point
  - 2 liens par socket CPU
  - 1 pour la connexion à l'autre socket
  - 1 pour la connexion au chipset
  - Jusqu'à 6.4 GT/Sec dans chaque direction
- QuickPath Memory controller (DDR3) intégré



Nehalem

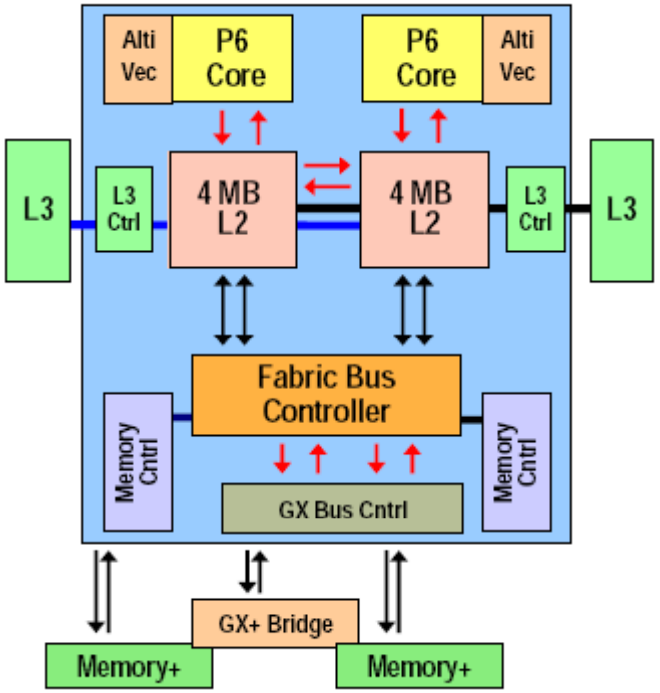
Example Platform Topologies



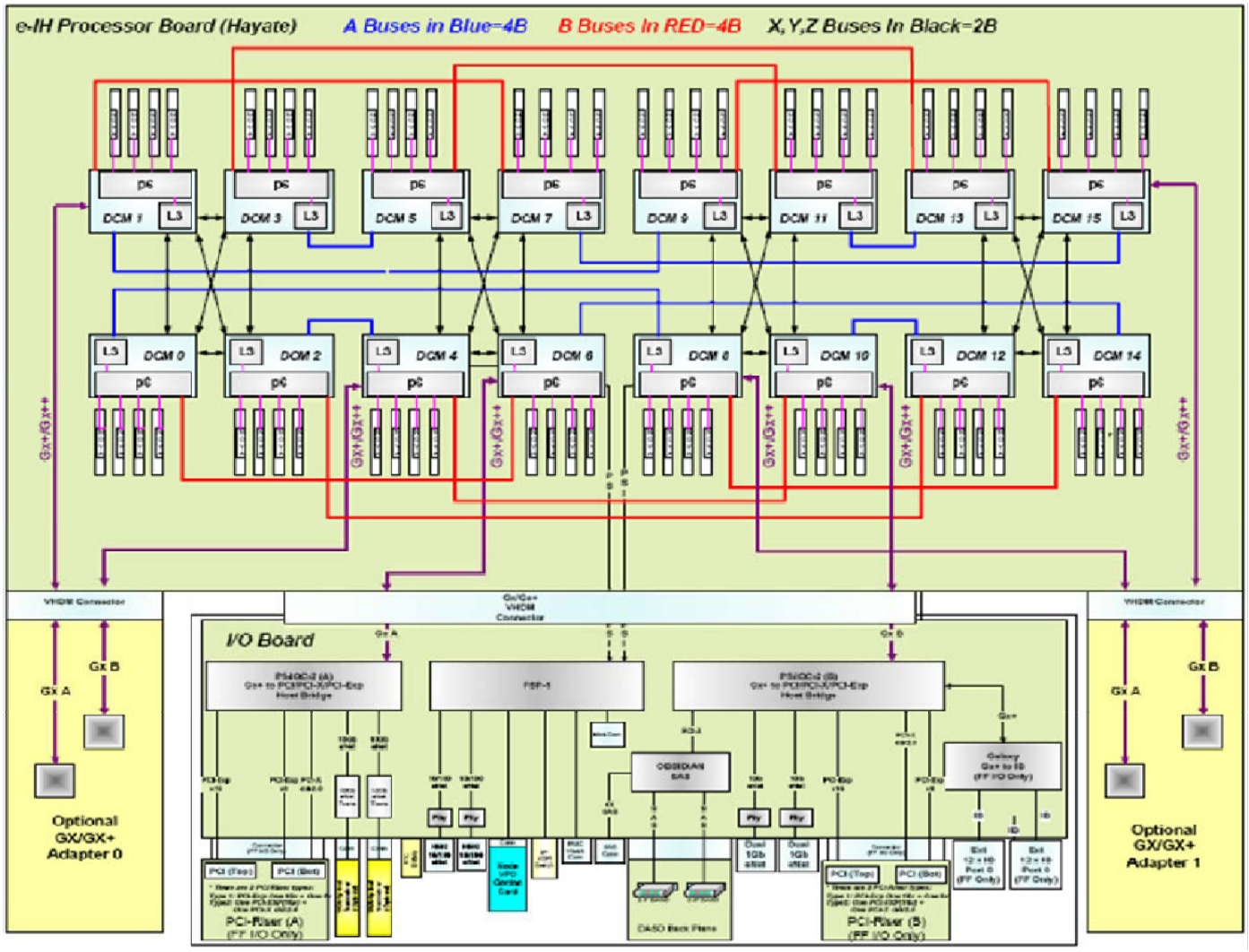
Intel® QuickPath Interconnect Links Shown in Orange Arrows

Architecture Power6

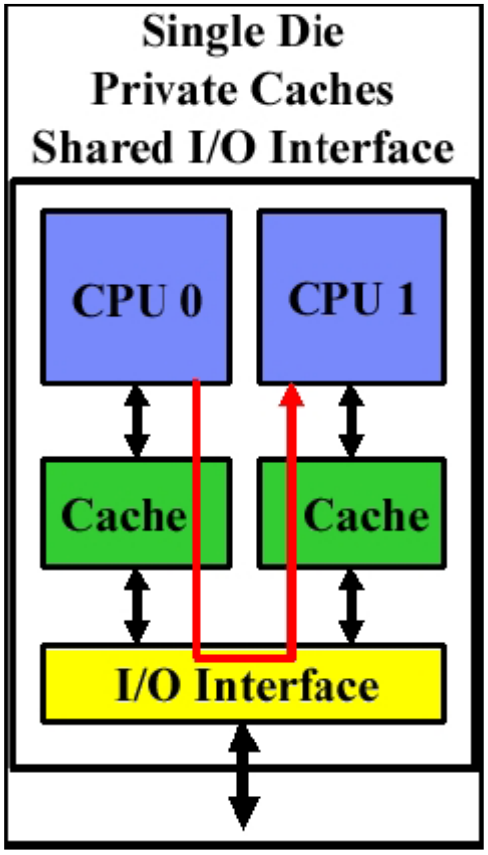
- Chip dual core haute fréquence (↗4.7 Gz)
- 2 voies SMT core
- 8 unités d'exécution : (2 L/S, 2 FP, 2 FX, 1 BR, 1 VMX)
- 790 M de transistors, 341 mm^2 par die (comparable au power5)
- Jusqu'à 64 cores par système SMP (switch d'interconnection permettant de relier 32 power6)
- Cache L2 de 4 Mo /core
- Un contrôleur du cache L3 et mémoire sur le chip
- Cache L3 de 32 Mo
- Jusqu'à 100 Go/s de BP mémoire

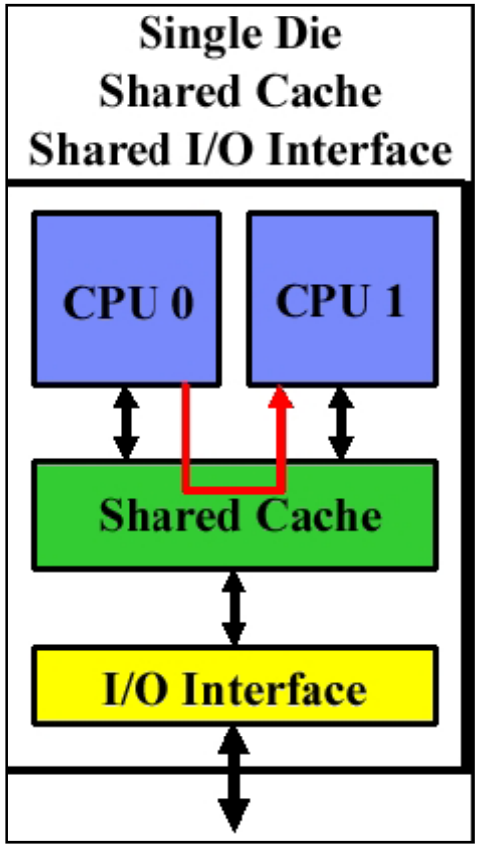


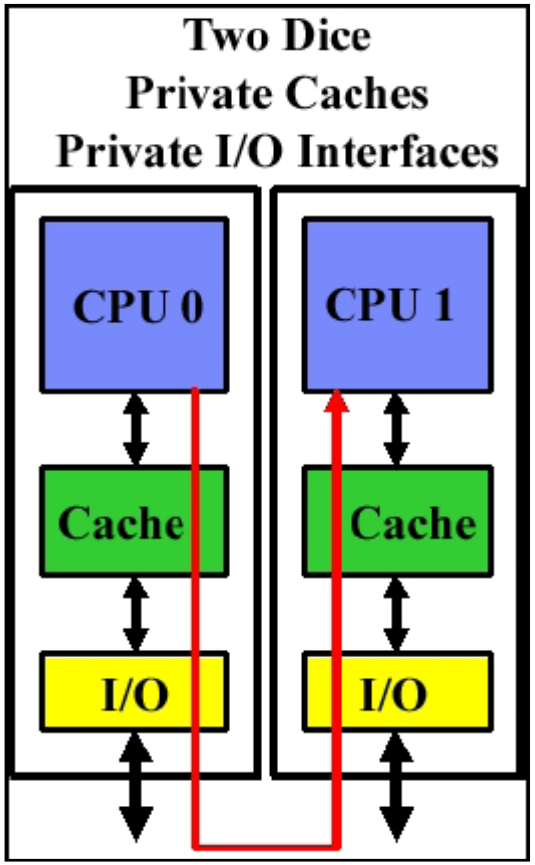
Système d'interconnexion interne des architectures P575 d'IBM



Architecture des caches





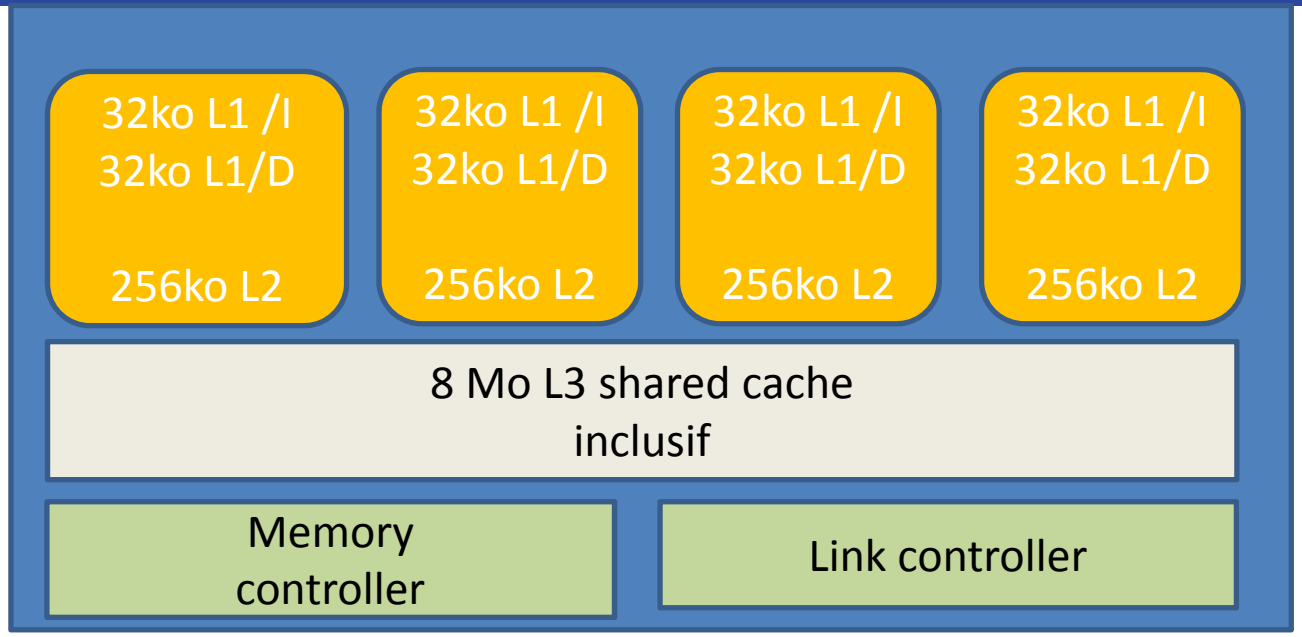


Partage des caches L2 et L3

- Partage du cache L2 (ou L3) :
  - ☺ communications plus rapides entre coeurs,
  - ☺ meilleure utilisation de l'espace,
  - ☺ migration des threads plus facile entre les coeurs,
  - ☹ contention au niveau de la bande passante et des caches (partage de l'espace),
  - ☹ problème de cohérence.
- Pas de partage entre les caches :
  - ☺ pas de contention,
  - ☹ communication/migration plus coûteuse, passage systématique par la mémoire.

  - cache L2 privé, cache L3 partagé :IBM Power5+ / Power6, Intel Nehalem
  - tout privé : le Montecito

Exemple du Nehalem
Une hiérarchie de cache à 3 niveaux



- Cache de niveau 3 inclusif de tous les niveaux précédent

4 bits permettent d'identifier ds quel cache de quel proc se trouve la donnée

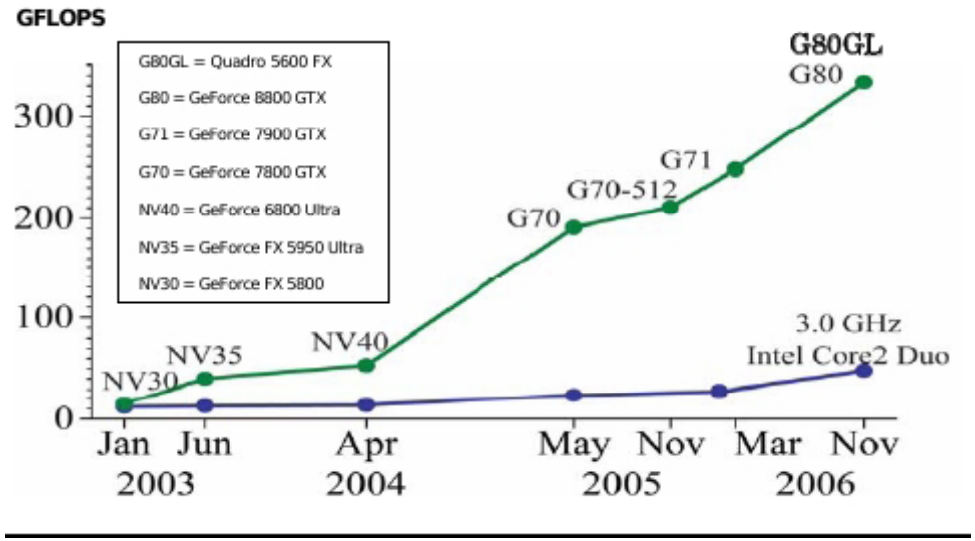
  - ☺ limitation du traffic entre cœurs
  - ☹ Gaspillage d'une partie de la mémoire cache

Fonctionnement différent de son concurrent le Barcelona

Mais technologie HT assist sur le processeur Istanbul

Les processeurs graphiques

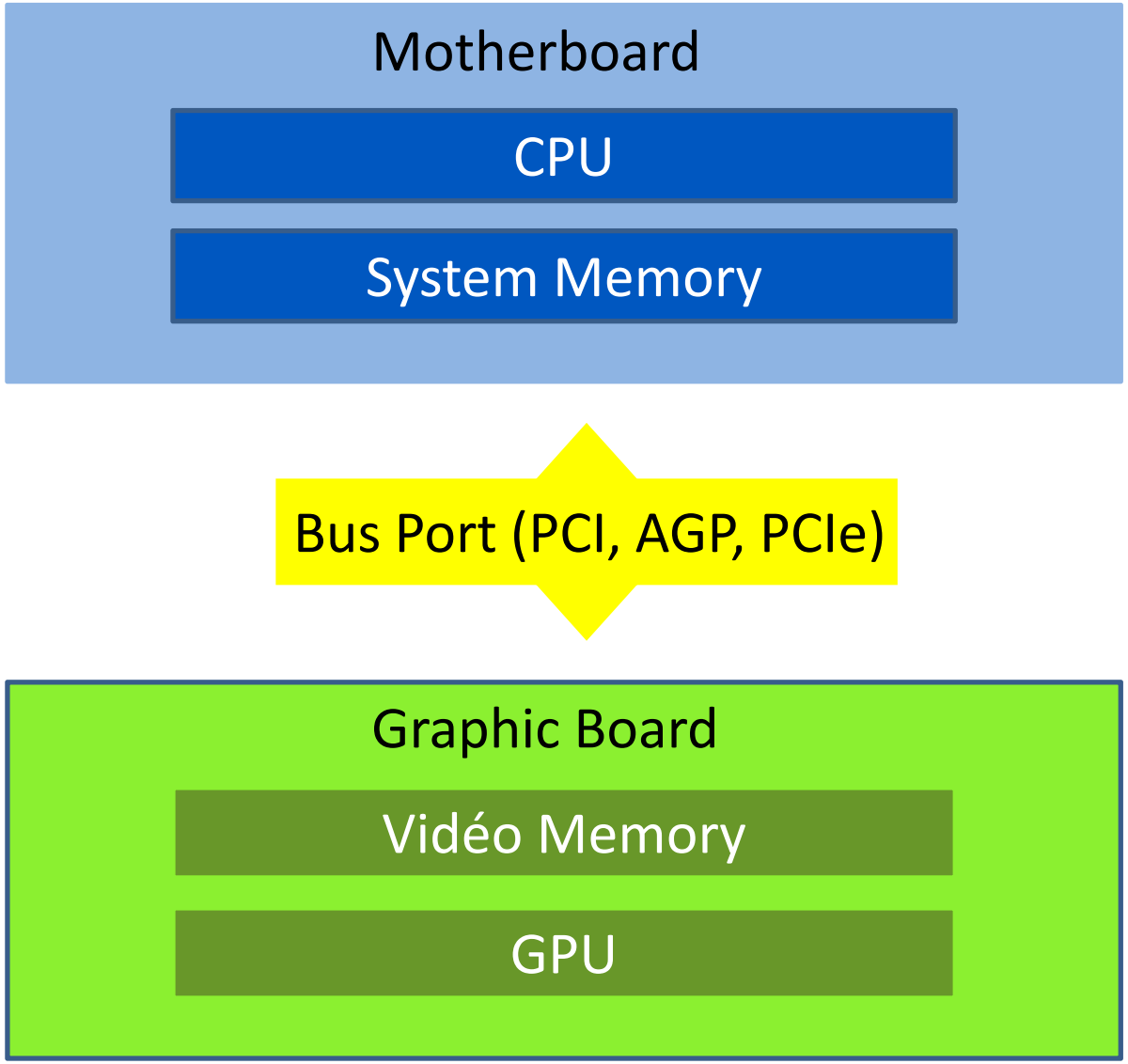
GPU = Graphic Processor Unit



- Performance théorique GeForce 8800GTX vs Intel Core 2 Duo 3.0 GHz : 367 Gflops / 32 GFlops.
- Bande passante mémoire : 86.4 GB/s / 8.4 GB/s.
- Présents dans tous les PC : un marché de masse.
- Adapté au massivement parallèle (milliers de threads par application).
- Jusqu'à récemment, uniquement programmable via des APIs graphiques.

Aujourd'hui, des modèles de programmation disponibles : CUDA (Compute Unified Device Architecture).

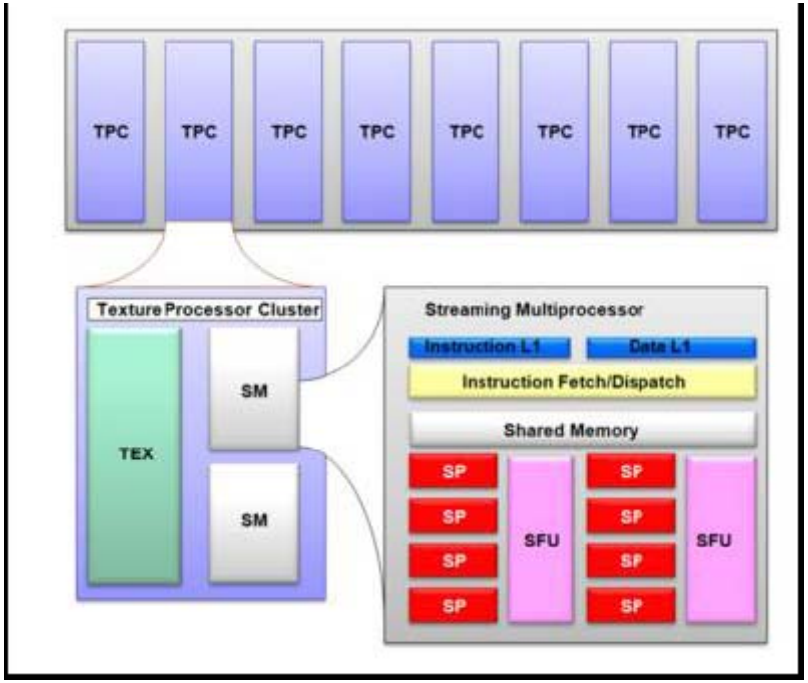
Architecture de PC classique



Exemple : GeForce 8800

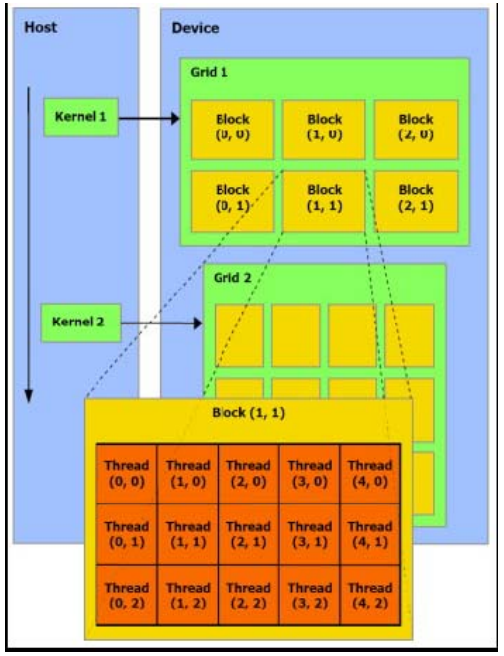
128 Unités de calcul !
8 partitions de 16

Architecture GeForce8800

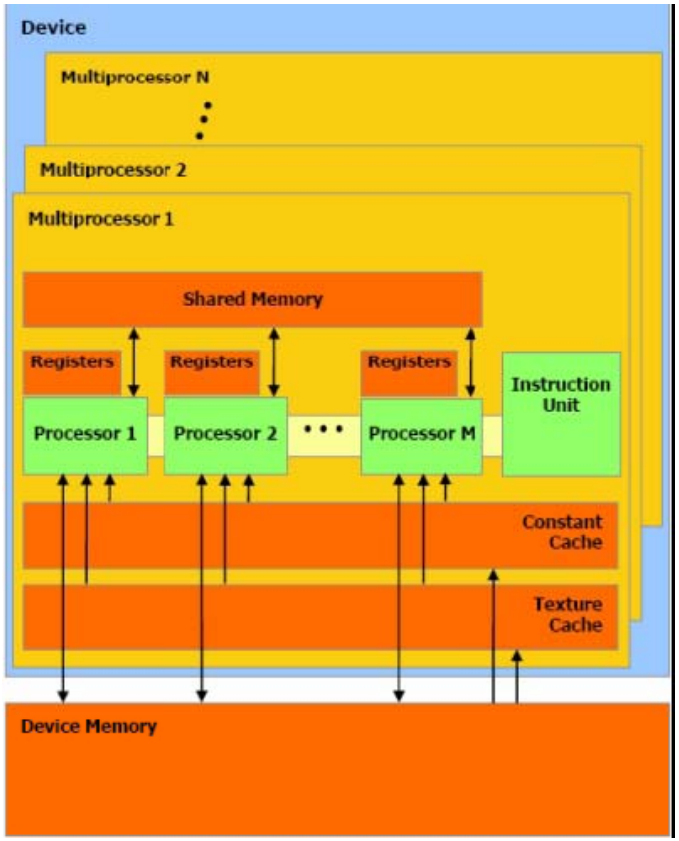


- Architecture de type SIMD (Single Instruction Multiple Data).
- Peut être vu comme une grosse unité de calcul divisée en multiprocesseurs.
- Chaque multiprocesseurs est composé de plusieurs Stream Processors.

- SIMD ⇨ un même programme (kernel) exécuté plusieurs fois simultanément.
- Chaque exécution = 1 thread (élément de base).
- Threads groupés en warps (32 threads) à l'intérieur de blocs (64 à 512 threads) ; les 32 threads du warp sont exécutées en parallèles.
- Les threads d'un bloc peuvent coopérer.
- Ces blocs sont réunis dans des grilles.



Architecture Geforce8800



- Niveau GPU : Beaucoup de RAM (> 128 Mo).
- Niveau Multiprocesseur : Mémoire partagée (16 ko), accès très rapide, permet la communication entre threads. Des caches (constante et texture).
- Chaque Stream Processor : des registres (8192 * 32 bits).

Des modèles de GPU plus récents

- Geforce GTX 280 de NVIDIA
  - Puissance de calcul brute : 933 Gflops
  - Nouveauté majeure : **la double précision**
  - Doublement de la taille des registres
  - 240 processeurs de flux (30 Multi-processeurs), seule une trentaine fonctionne en double précision
  - Amélioration de la bande passante mémoire
- Tesla C1060 NVIDIA, équivalent avec une mémoire de 4 Go au lieu de 1 Go.
  - Puissance de calcul brute : 933 Gflops simple précision
    78 Gflops double précision
  - 240 processeurs de flux
  - Débit mémoire Max : 102 Go/s
- ATI/AMD Firestream 9270
  - Puissance de 1.2 Tflops en SP, 200 Gflops en DP

Comparaison GPU /CPU

A performance intrinsèque égale :

Les plateformes à base de GPUs :

- occupent moins de place
- sont moins chères
- sont moins consommatrices d'électricité

Cf exposé NVIDIA aux journées Teratec

CUDA Nvidia

- Ensemble d'extensions au langage C et une API.
- Mot clé principal : __global__, placé devant une fonction, indique qu'elle est un kernel, cad une fonction qui va être appelée par le CPU et exécutée par le GPU. Il faut lui indiquer la taille de la grille et la taille de chaque bloc sur lequel le kernel est appliqué.
- __device__ pour sa part désigne une fonction qui sera exécutée par le GPU mais qui n'est appelable que depuis le GPU.
- L'API CUDA offre quant à elle essentiellement des fonctions de manipulation mémoire en VRAM.

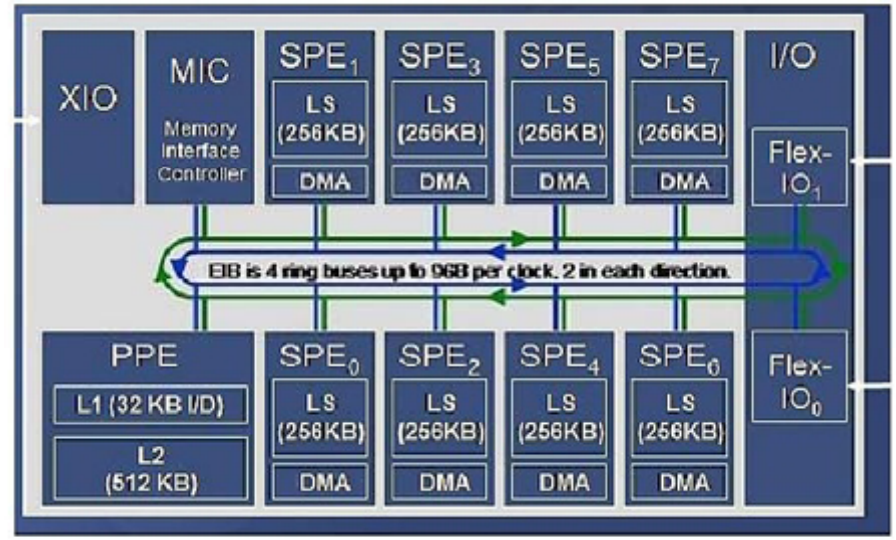
Programmation CUDA

Optimiser un programme :
équilibrer au mieux le nombre de blocs et leur taille : plus de threads par blocs permet de masquer la latence des opérations mémoires mais diminue le nombre de registres disponibles par threads.

Autres langages

- HMPP
  - Expression du parallélisme et des communications dans le code C ou Fortran à l'aide de directive à la « OpenMP »
  - HMPP gère les communications CPU-GPU
- Brook +
  - Principalement AMD
- OpenCL (Open Computing Language):
  - Un Standard en cours de finalisation
  - interface de programmation basée sur du C avec des extensions du langage
  - portable
  - calcul parallèle sur architecture hétérogène : CPUs, GPUs, et autres processeurs.

Processeur spécialisé : le CELL



- Développé par Sony, Toshiba et IBM, processeur de la PlayStation 3.
- Un processeur est composé de Un coeur principal (PPE) et 8 coeurs spécifiques (SPE).
- le PPE : processeur PowerPC classique, sans optimisation, « in order », il affecte les tâches aux SPEs
- les SPEs : constitués d'une mémoire locale (LS) et d'une unité de calcul vectoriel (SPU). Ils ont un accès très rapides à leur LS mais pour accéder à la mémoire principale ils doivent effectuer une requête de transfert asynchrone à un bus d'interconnexion. Les SPEs exécutent les tâches calculatoires.
- Le travail d'optimisation est à la charge du programmeur.

Parallélisme sur le CELL

- Les SPU permettent de traiter 4 op 32 bits/cycle (registre de 128 b).
- Programmation explicite des threads indépendantes pour chaque coeur.
- Partage explicite de la mémoire : l'utilisateur doit gèrer la copie des données entre cœurs ⇨ Plus difficile à programmer que les GPUs (car pour les GPUs, les threads ne communiquent pas entre différents multiprocesseurs, excepté au début et à la fin).

Processeur CELL : performance crête (registres 128b, SP)
4 (SP SIMD) x 2 (FMA) x 8 SPUs x 3.2 GHz = 204.8 GFlops/socket (en SP)
20.8 en DP

Processeurs spécialisés
Programmation hybride

- FPGA (Field Programmable Gate Array)
  - Trop spécialisé, adapté à des problèmes très spécifiques
- CELL
  - architecture intéressante mais difficile à programmer
- GPU
  - de plus en plus performant
  - des outils pour les programmer en plein développement
  - disponibles partout, pas chère

Mais adapté à un modèle SIMD et à un //isme massif
Le GPU en tant que co-processeur (architecture hybride) offre de nouvelles perspectives, introduit de nouveaux modèles de programmation

Technologie réseau pour le HPC

Technologie	Vendeur	Latence MPI usec, short msg	Bande passante par lien unidirectionnel, MB/s
NUMAlink 4	SGI	1	3200
QsNet II	Quadrics	1.2	900
Infiniband	Mellanox, Qlogic, Voltaire	1.O7	2500 (SDR) Double speed DDR Quad speed QDR
High Performance Switch	IBM	5	1000
Myri-10G	Myricom	2.3	2500
Ethernet 10 Gb		< 10	2600
Ethernet 1 Gb		> 40	50 à 80

Réseau Infiniband

• Réseau haute performance très flexible :
Liaison entre serveurs de données, entre baies de stockage, ou interconnexion des nœuds d'une grappe de calcul.
• Standard : # interfaces de différents constructeurs, interopérables

Emploi du RDMA (Remote Direct Memory Access).
Les cartes s'utilisent au travers d'une pile logicielle OFED qui comprend un module noyau et une bibliothèque niveau utilisateur.

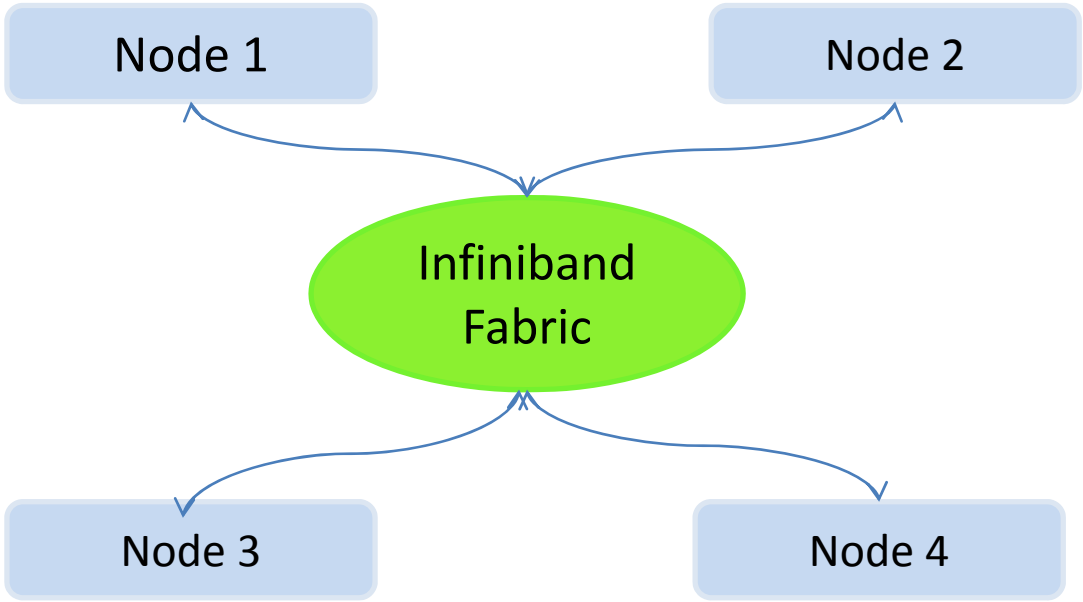
Réseau Infiniband

Plusieurs interfaces d'accès offertes par OFED :

- *verbs* : commandes élémentaires envoyées à la carte inifiniband, accès au RDMA
- DAPL : interface de + haut niveau. kDAPL (version noyau) + uDAPL (version utilisateur)
- SDP : émulation socket sur infiniband
- SRP : protocole d'accès à des disques SCSI distants par infiniband.
- IPoIB (IP sur IB) : encapsulation de paquets IP sur Infiniband.

DAPL, SDP, SRP et IPoIB protocoles de plus haut niveau construits au dessus des *verbs*.

Réseau Infiniband – Topologie de base



Le fabric peut être un switch, ou un ensemble de plusieurs switchs interconnectés et de routeurs.
Le node peut être un serveur, un périphérique d'entrées/sorties tel qu'un système RAID.
Chaque connexion est une connexion point à point, la capacité du réseau est entièrement dédiée de bout en bout.
GUID ou GID : Global Unique ID LID : Local ID (locale au subnet)

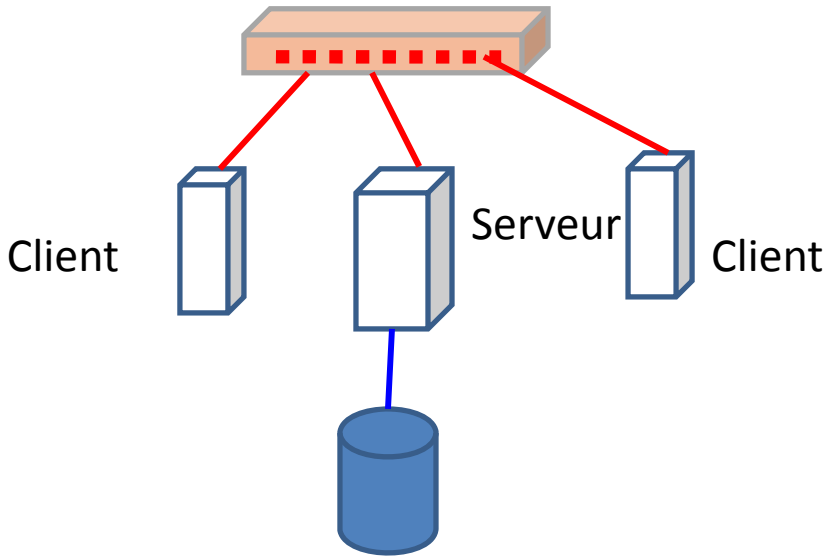
Le système d'entrées/sorties

- Le système de fichier local

  Ex: ext2, ext3, Fat, xfs, ntfs, …

Partage des données :

- Système de fichiers distribué : partage sur le réseau en mode client/serveur Ex : NFS



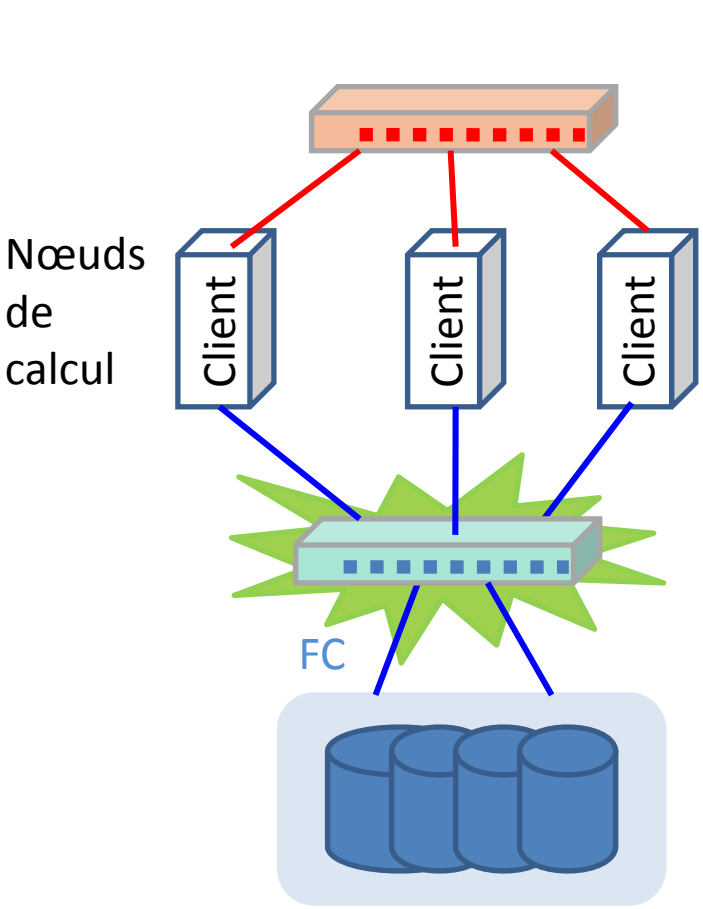
1 seul serveur

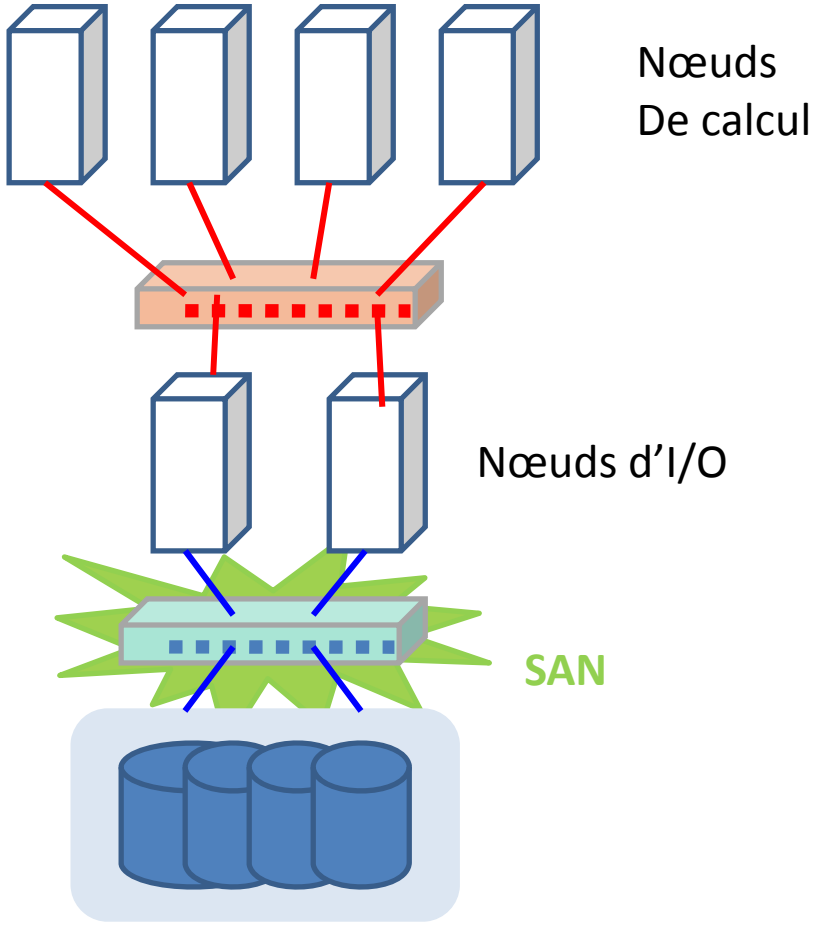
⇒ Limite le nombre de clients (quelques dizaines)

Le système d'entrées/sorties

- Storage Area Network (SAN) File System : un ensemble de machines partagent du stockage sur Fiber Channel

  Ex: CXFS, GPFS, StorNext



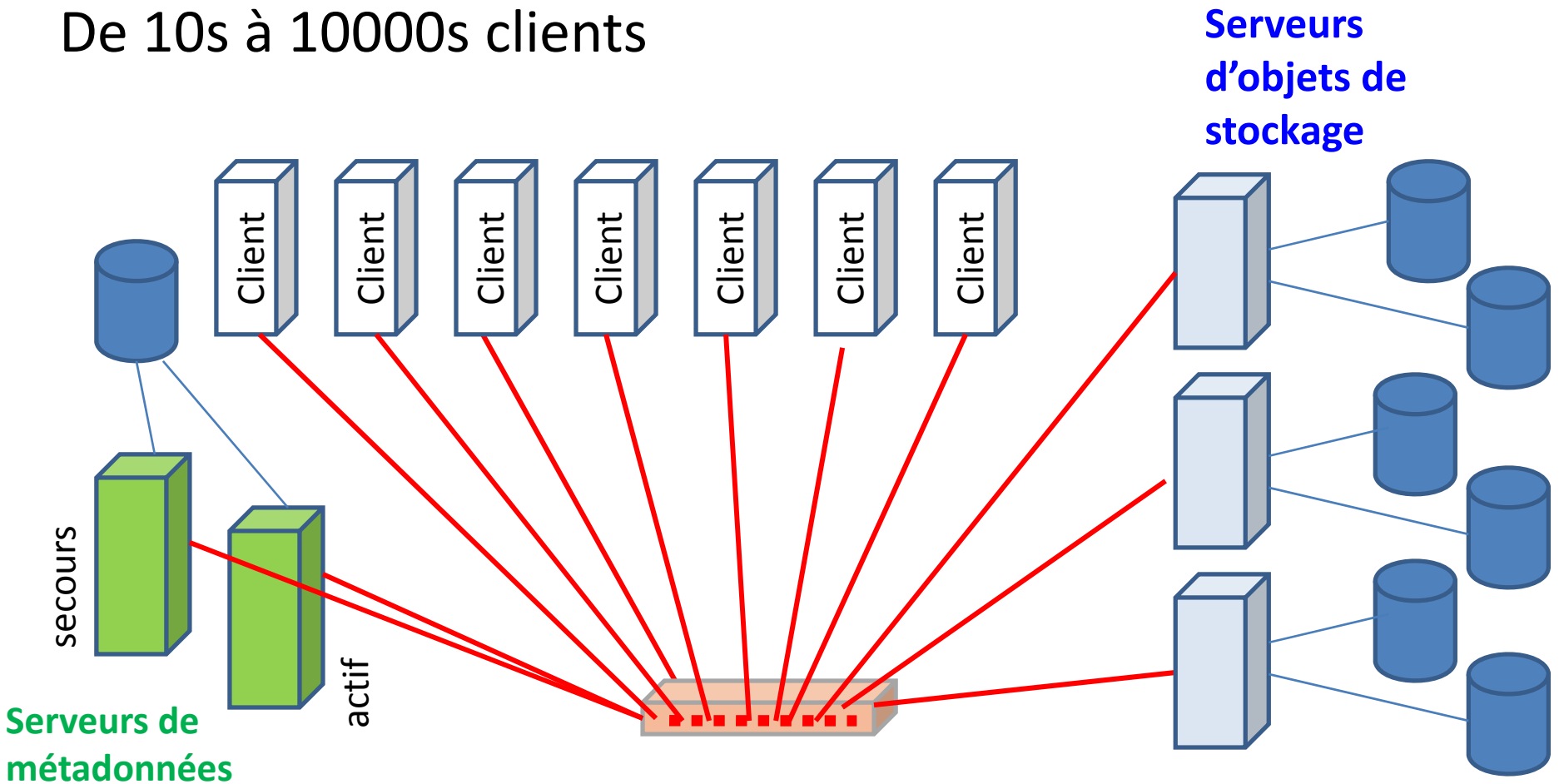


Le système d'entrées/sorties

- Parallel File System : plusieurs serveurs assurent le transport des données vers de multiples clients

Ex: Lustre, Panasas

De 10s à 10000s clients



Entre la théorie et la pratique …

Fréquence d'horloge, nombre de cœurs, nombre d'unités fonctionnelles, fréquence et capacité mémoire, caches, stockage, interconnexions réseau interne et externe …

Tous les éléments ont leur importance ; l'architecture doit être équilibrée pour que l'ensemble soit performant

- Les aspects logiciels sont très importants :

Le support des composants par le système, la configuration du système, la gestion des processus, les compilateurs, …

- L'intégration des composants, leur interconnexion

Pour éviter les surprises, tester vos applications sur la machine cible avec les logiciels proposés pour identifier d'éventuels goulots d'étranglement

Conclusion

- Vers une augmentation du nombre de coeurs plutôt que de la fréquence d'horloge : problème de finesse de gravure, d'échauffement ...
- Programmation de plus en plus complexe :
  - Nécessité d'une bonne connaissance des architectures.
  - Pas de croissance des performances au niveau du coeur
  ⇨ parallélisation obligatoire pour un gain de performance.
  - Nécessité d'élaborer des algorithmes et des programmes capables d'exploiter ce grand nombre de processeurs.
  - Nécessité d'exploiter le parallélisme aux différents niveaux de hardware
  - Programmation sur des architectures hybrides avec différents types de processeurs.
- Exploitation des performances des processeurs graphiques : produire un code extrêmement parallélisé, capable d'utiliser TOUS les "threads", sinon on atteint que quelques % de la puissance disponible.

Références

Un certain nombre d'éléments de ces slides sont issus des présentations ou des documentations présentes sur les sites suivant :

- http://www.realworldtech.com
- http://www.ci-ra.org séminaire « trends in computer architecture » W. Jalby
- http://www.idris.fr séminaires de l'IDRIS
- www.intel.com
- www.nvidia.fr