Introduction aux Coprocesseurs

Jean-Baptiste Keck

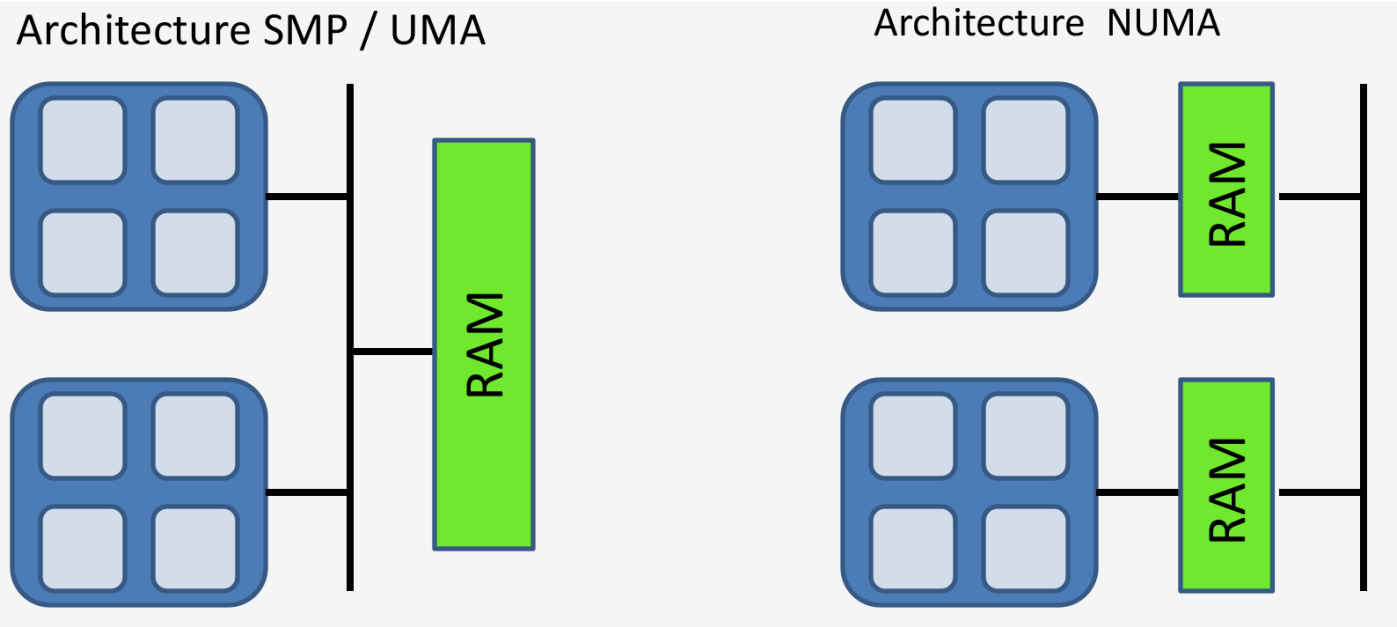
Laboratoire Jean-Kuntzmann - Grenoble

29 novembre 2017

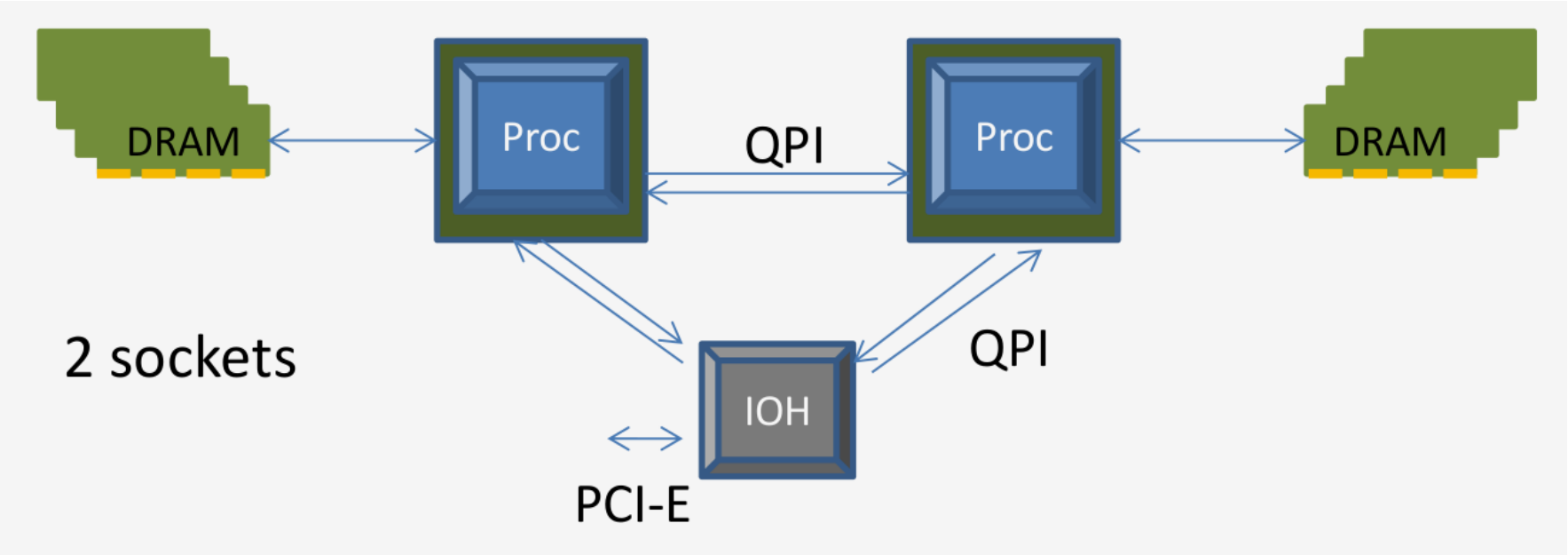
Table des matières

Modèles d'architectures partagées

- Architecture UMA (Uniform Memory Access) : Temps d'accès à un emplacement mémoire quelconque identique pour les les processeurs.
- Architecture NUMA (Non Uniform Memory Access) : Les processeurs peuvent accéder à la totalité de la mémoire les mais temps d'accès peuvent différer selon la distance coeur-mémoire.

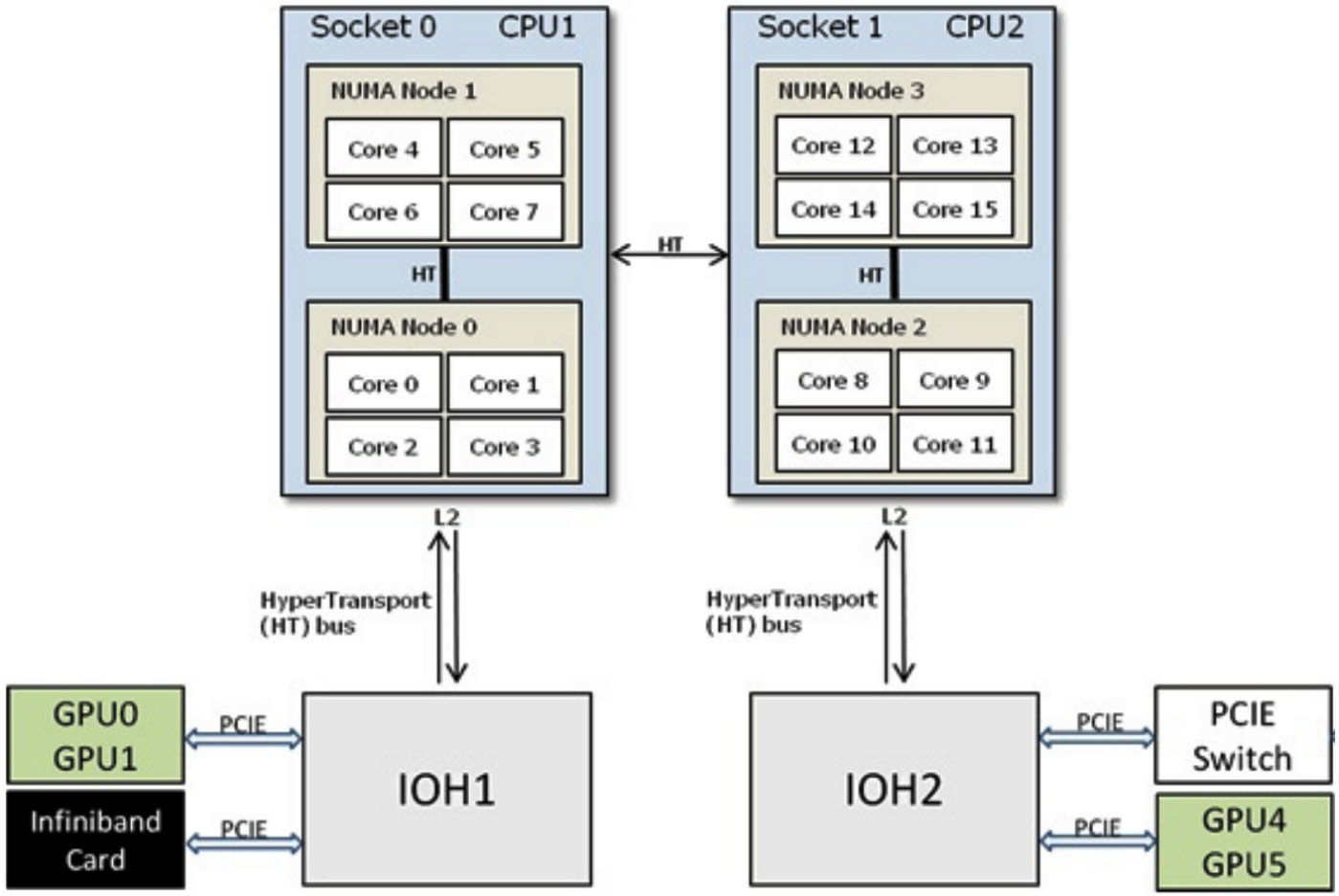


Rappel sur les noeuds de calcul

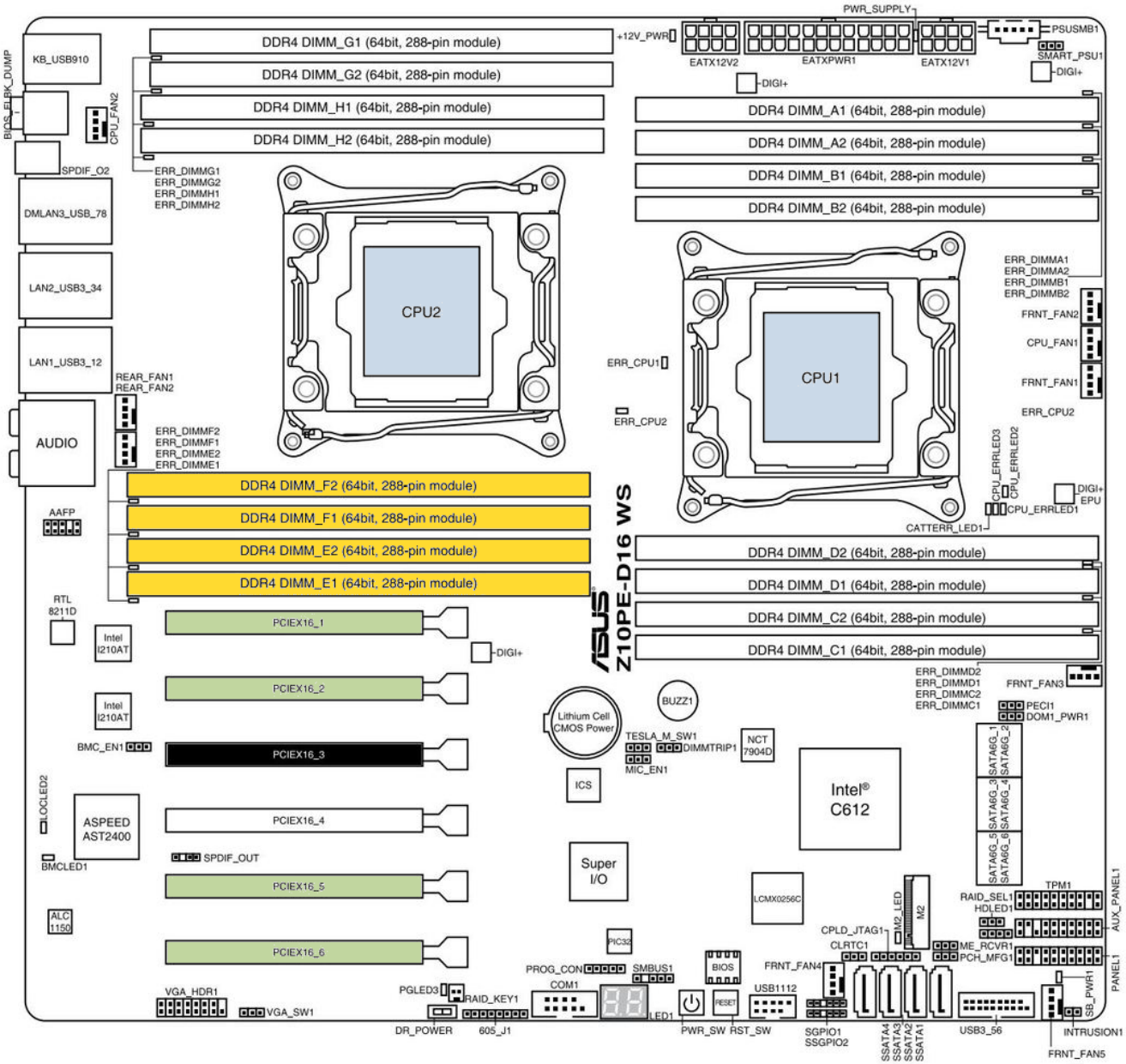


Noeud NUMA à deux sockets (deux processeurs physiques)

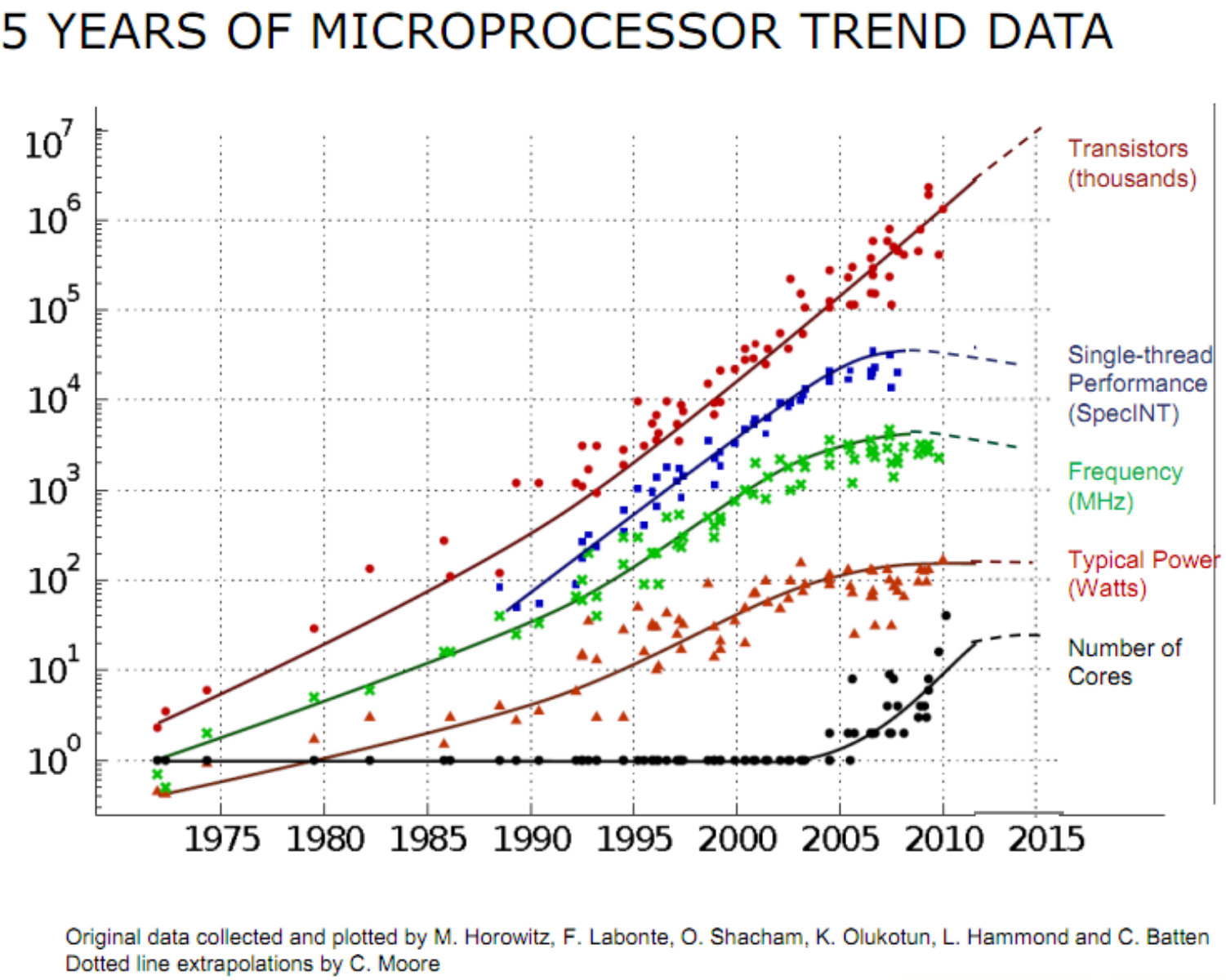
Vue globale d'un noeud de calcul (dual socket)



Carte mère dual socket

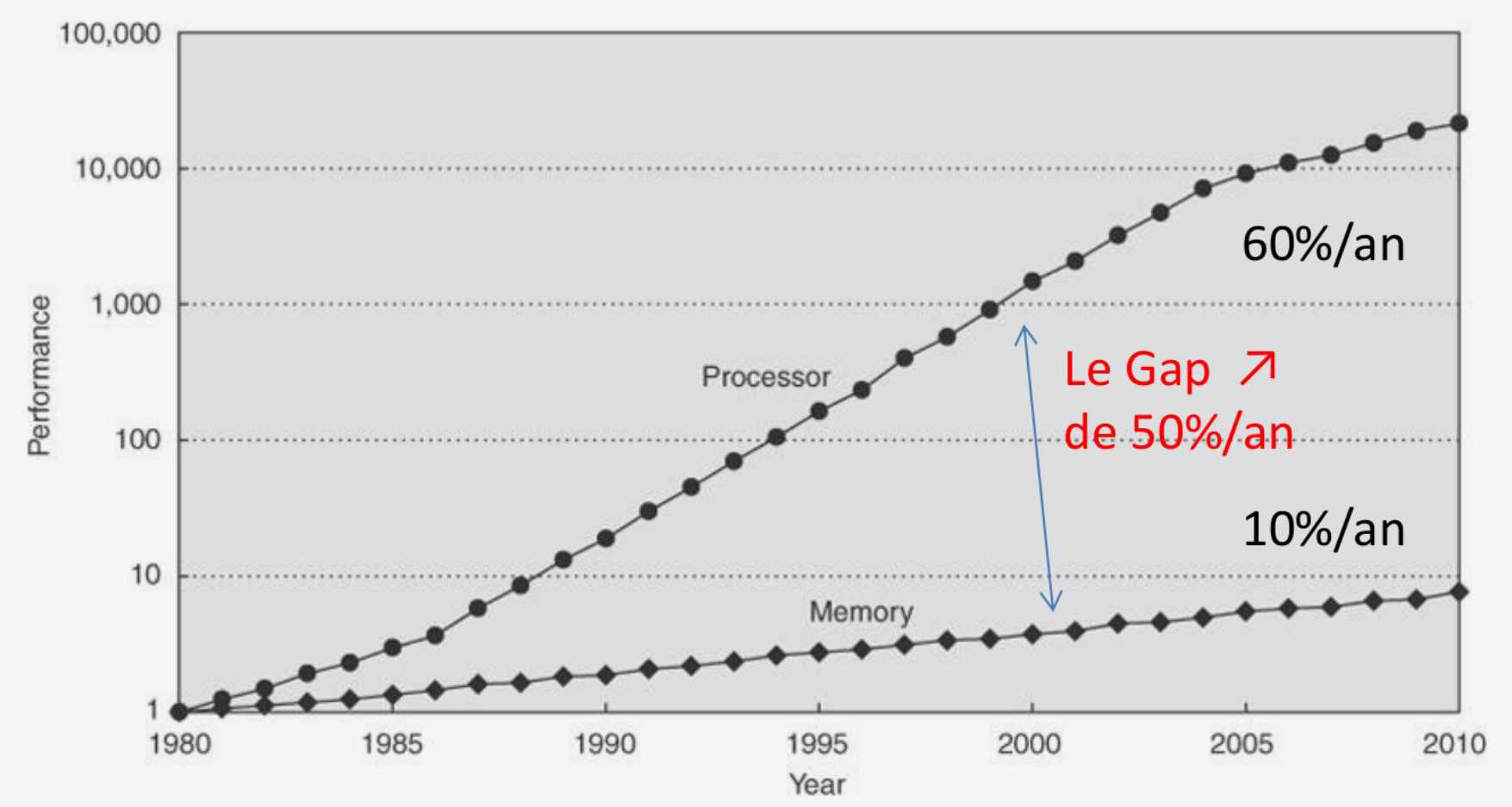


Quelques chiffres sur les CPU depuis les années 70

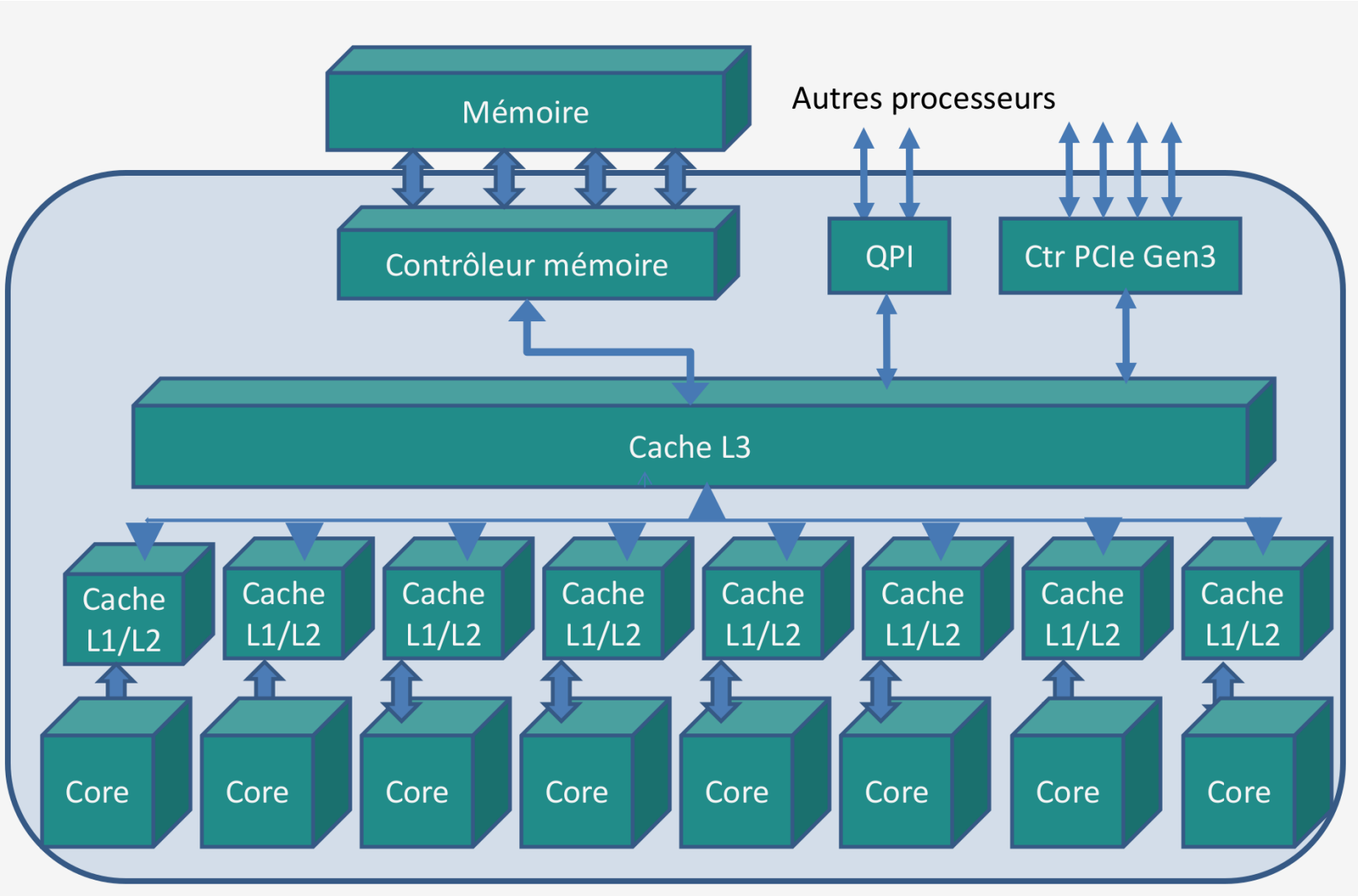


Performance processeur vs performance mémoire

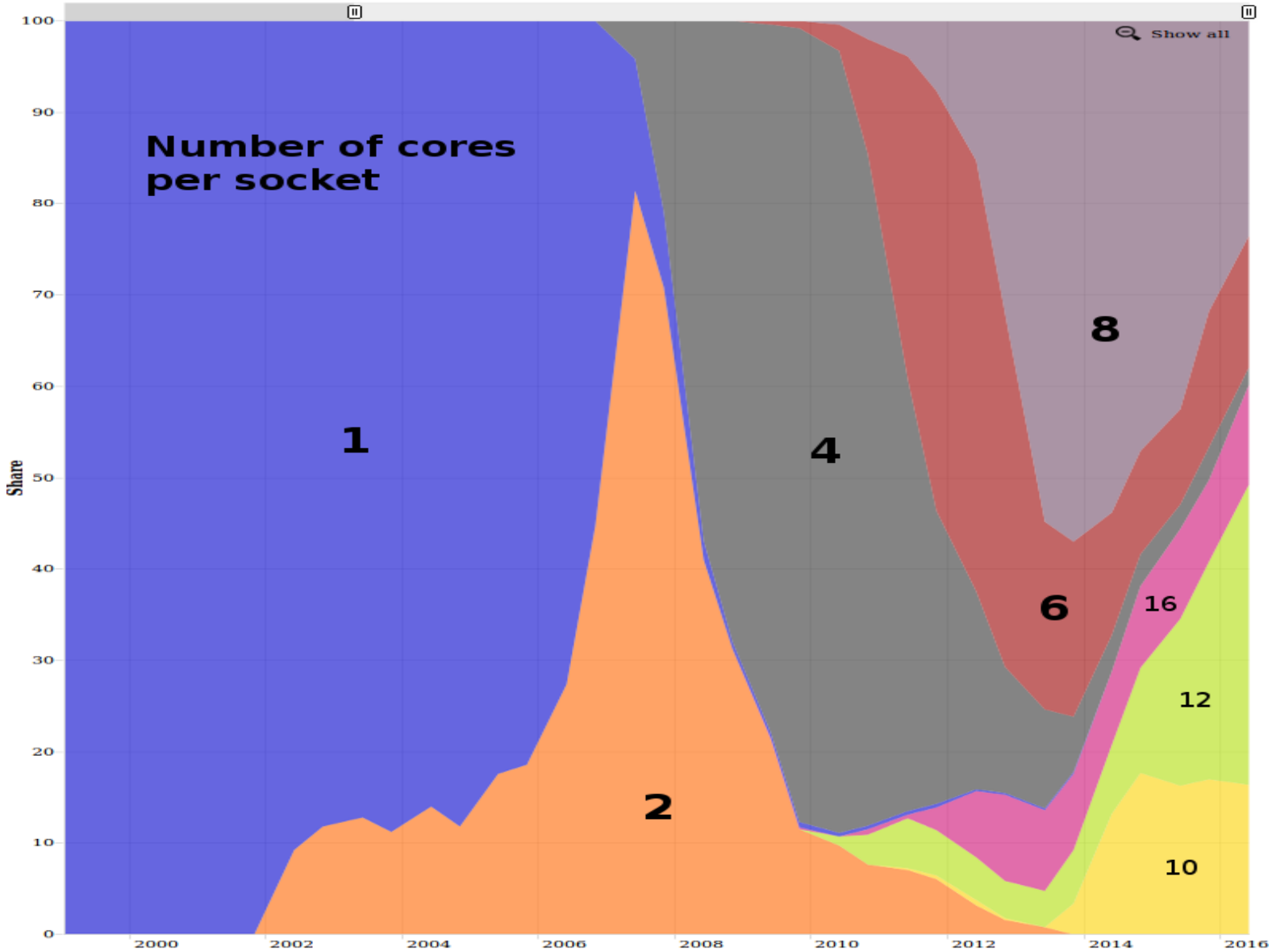
La performance des ordinateurs est surtout limitée par les accès mémoire (temps d'accès mémoire >> temps d'un cycle processeur).



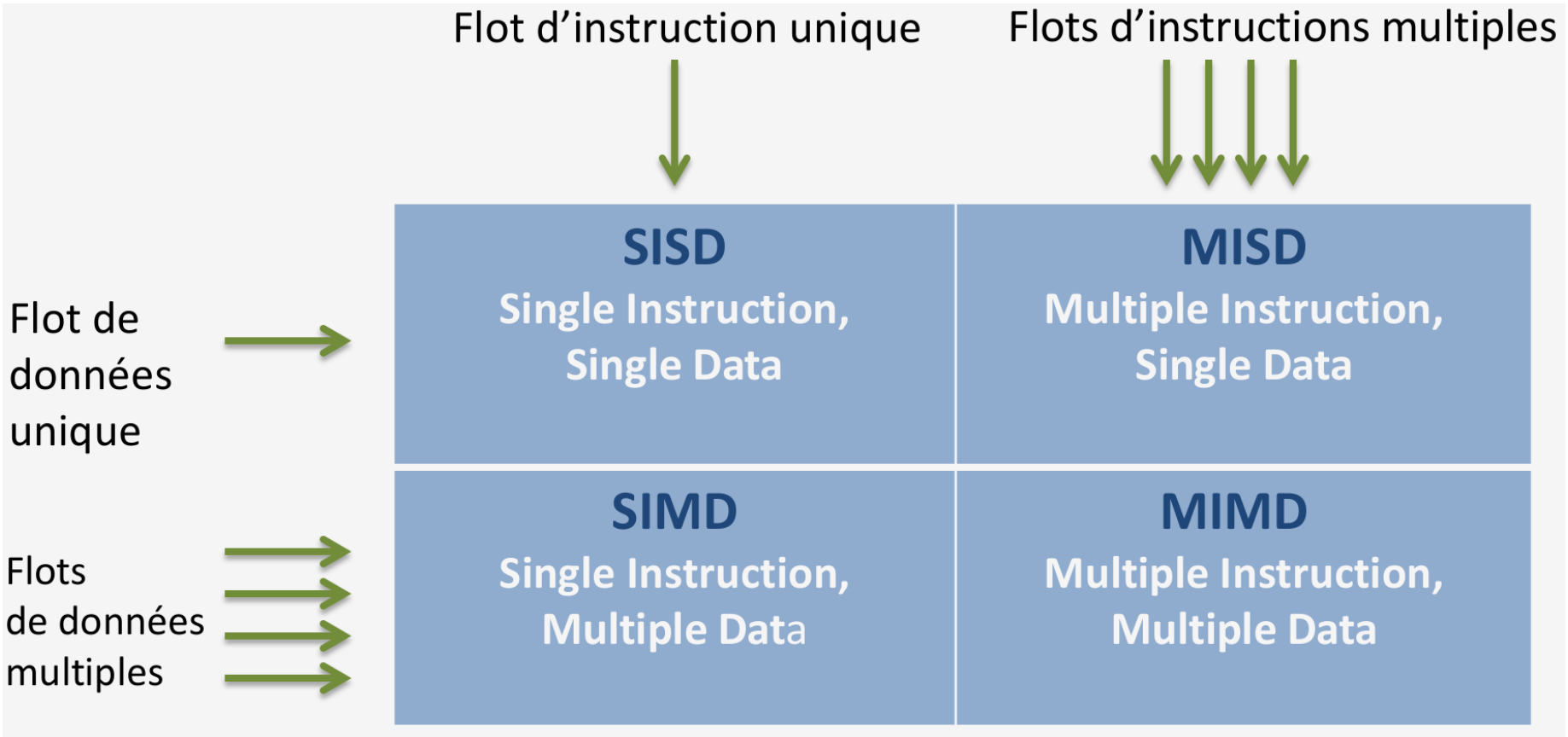
Un processeur en 2017



La tendance dans les supercalculateurs (TOP500)



Classification des machines parallèles (Flynn)



Execution d'une instruction

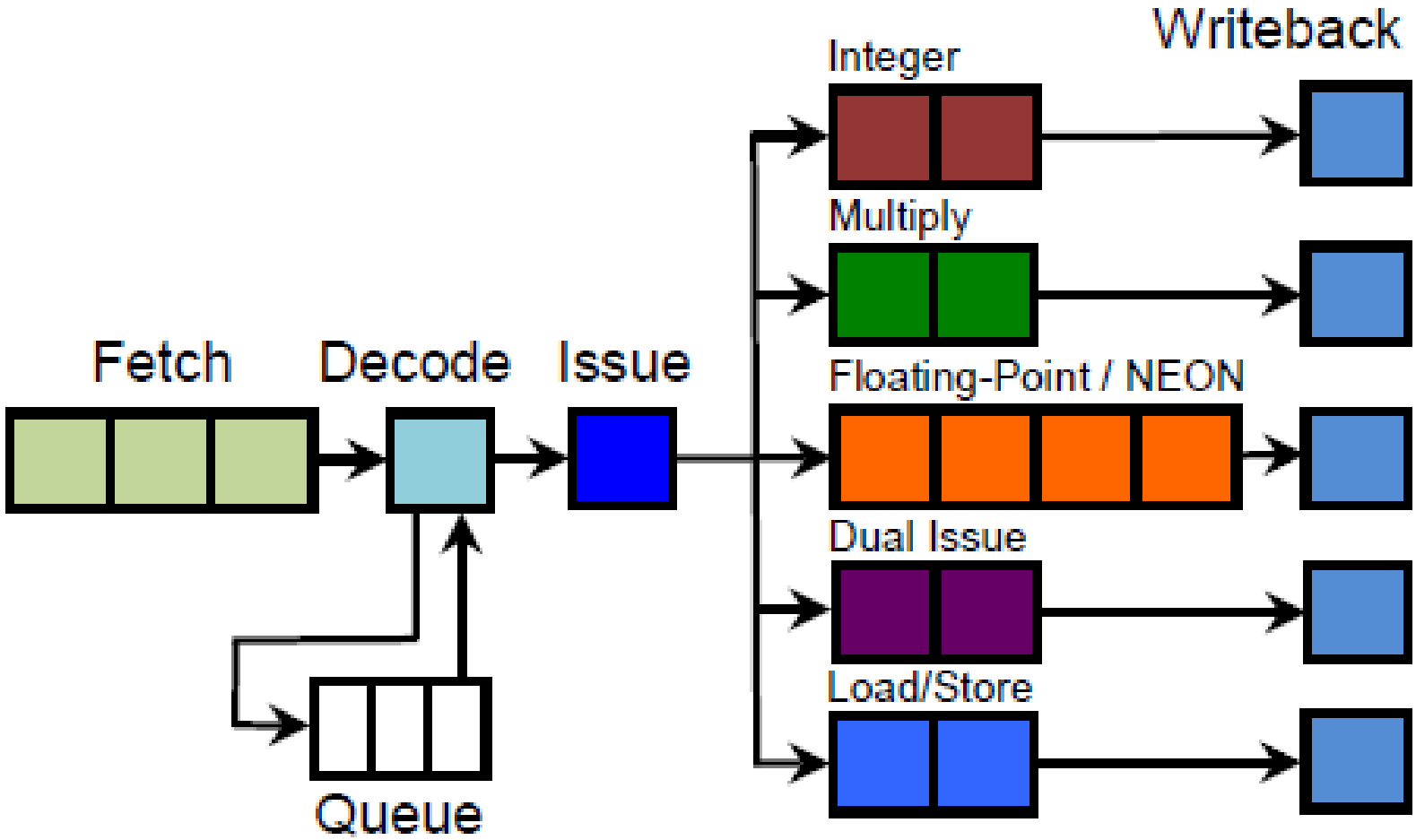
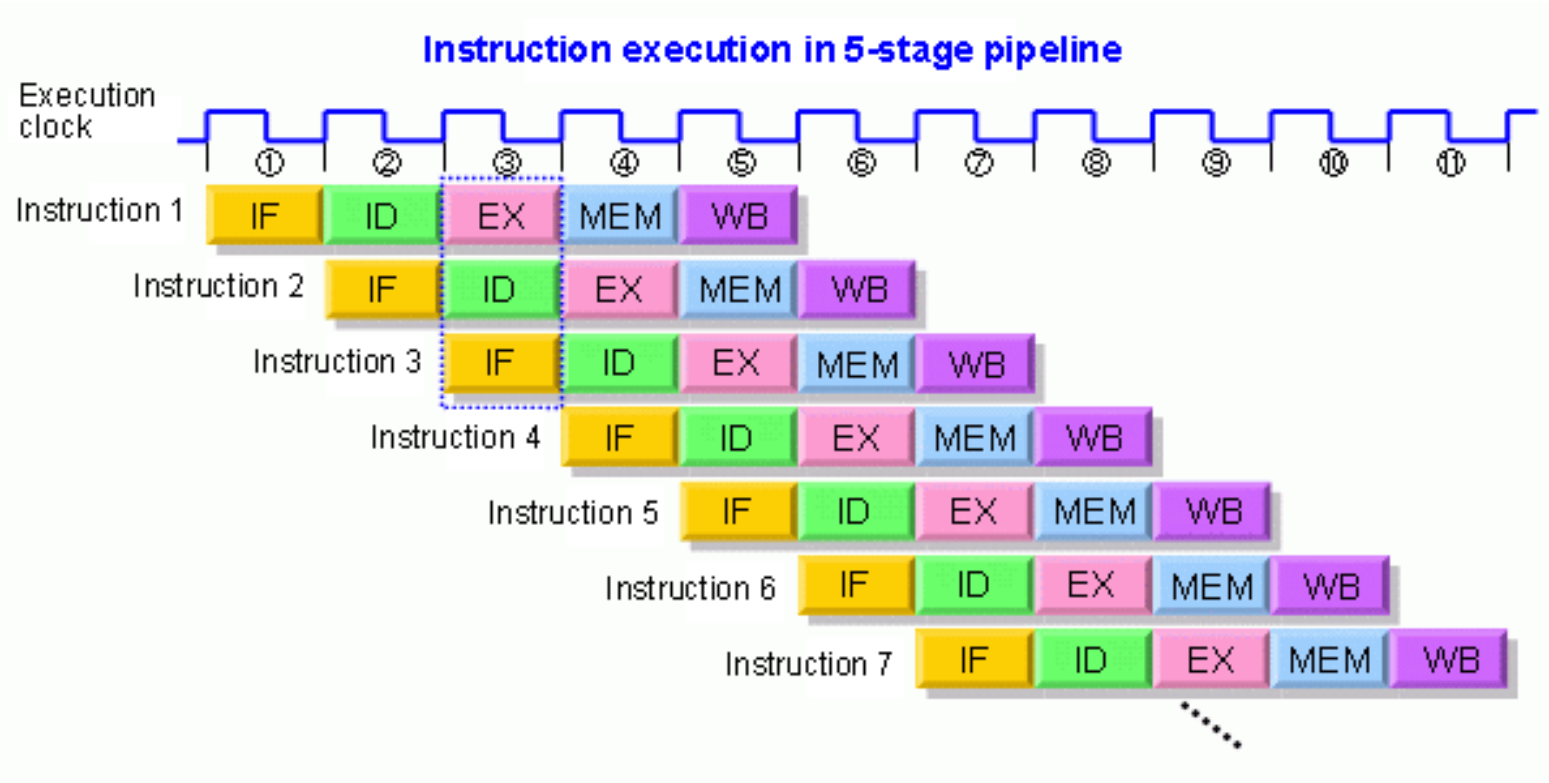


Figure 1 Cortex-A7 Pipeline

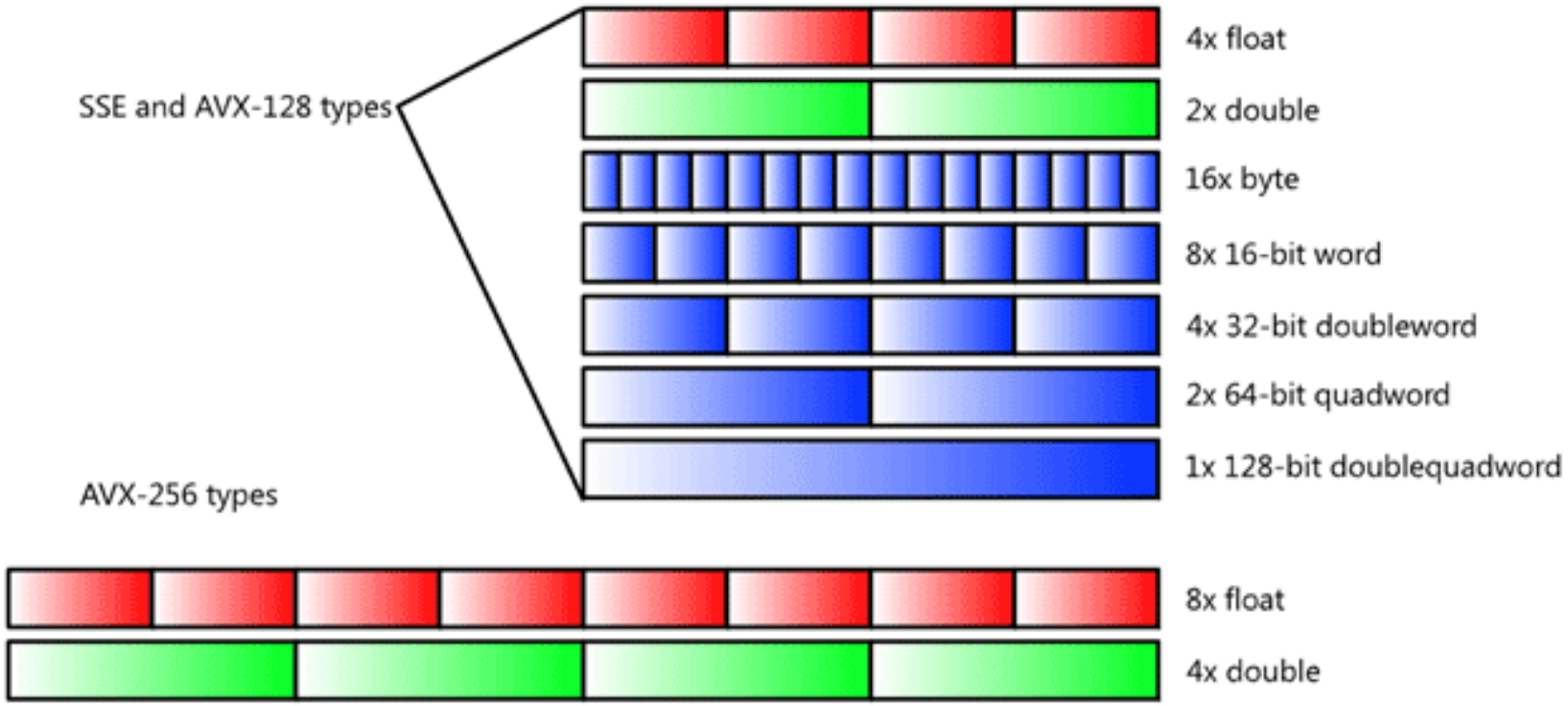
Pipelining : Multiplication des unités matérielles



- Execution dynamique : Le hardware modifie l'ordre des opérations pour favoriser l'occupation des ressources.
- Execution spéculative : Le hardware fait une hypothèse sur la suite d'un traitement après un branchement conditionnel.

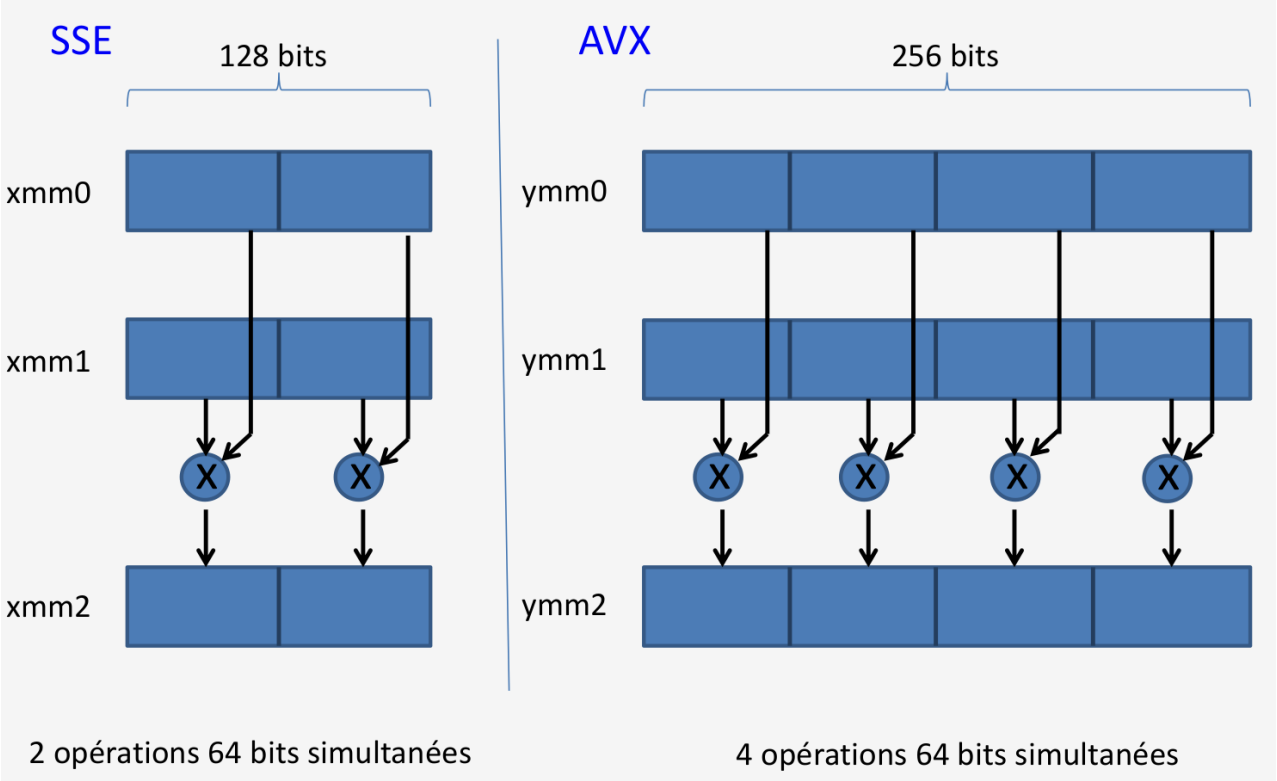
Parallélisme de données

- Instructions superscalaires : Possibilité d'exécuter une opération sur des registres vectoriels.
- Implémentations : AVX (Advanced Vector Extensions), SSE



Parallélisme de données

- Instructions superscalaires : Possibilité d'exécuter une opération sur des registres vectoriels.
- Implémentations : AVX (Advanced Vector Extensions), SSE

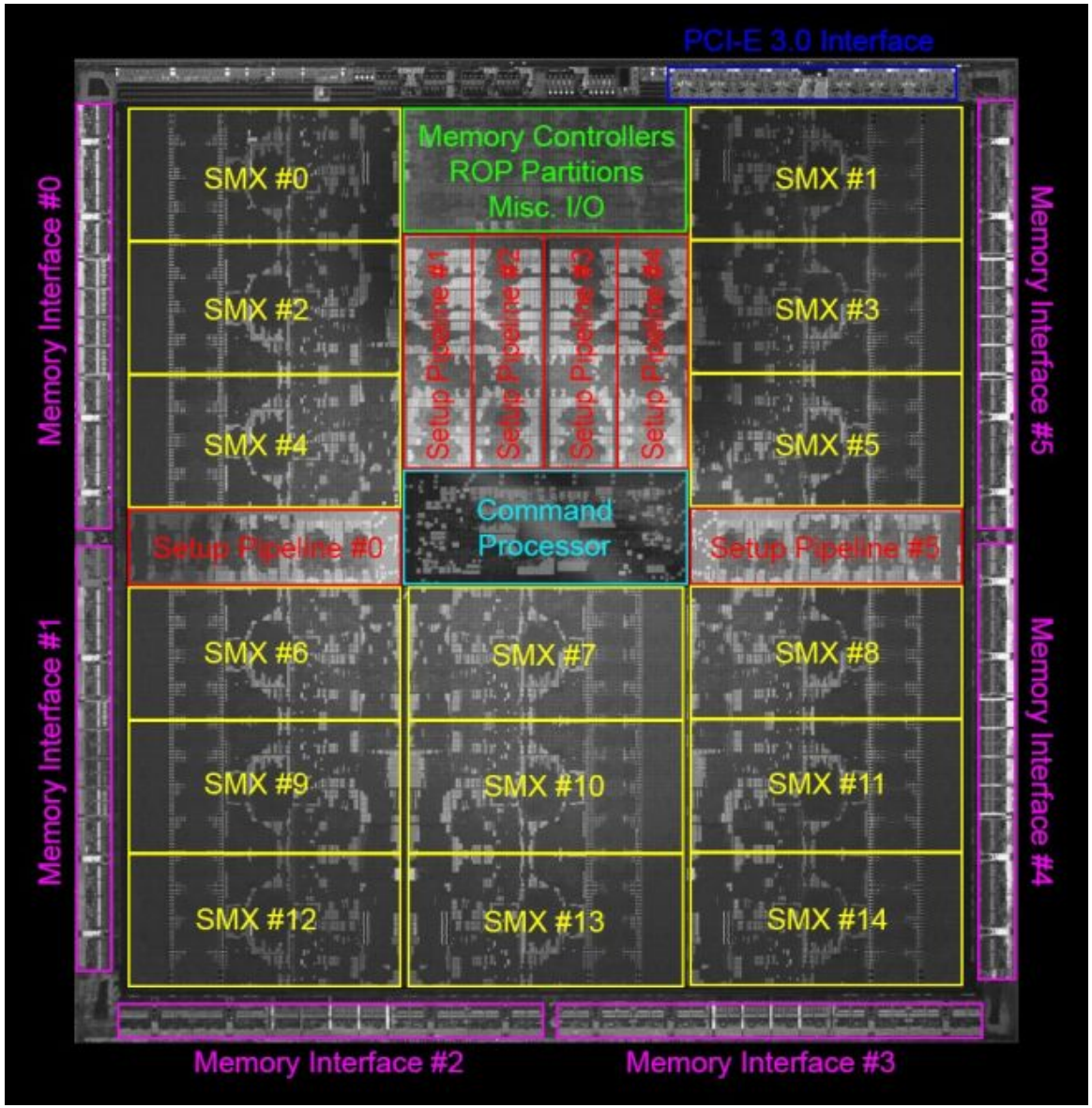


1. Rappels sur les noeuds de calcul et les processeurs
2. Notions de parallélisme
3. Les différents coprocesseurs existants
   - Motivations des coprocesseurs
   - Les GPU (Nvidia et AMD)
   - La réponse d'Intel : Les MIC
   - Parts des systèmes avec coprocesseur dans le TOP500
4. Caractériser un coprocesseur
5. Modèles de programmation des coprocesseurs

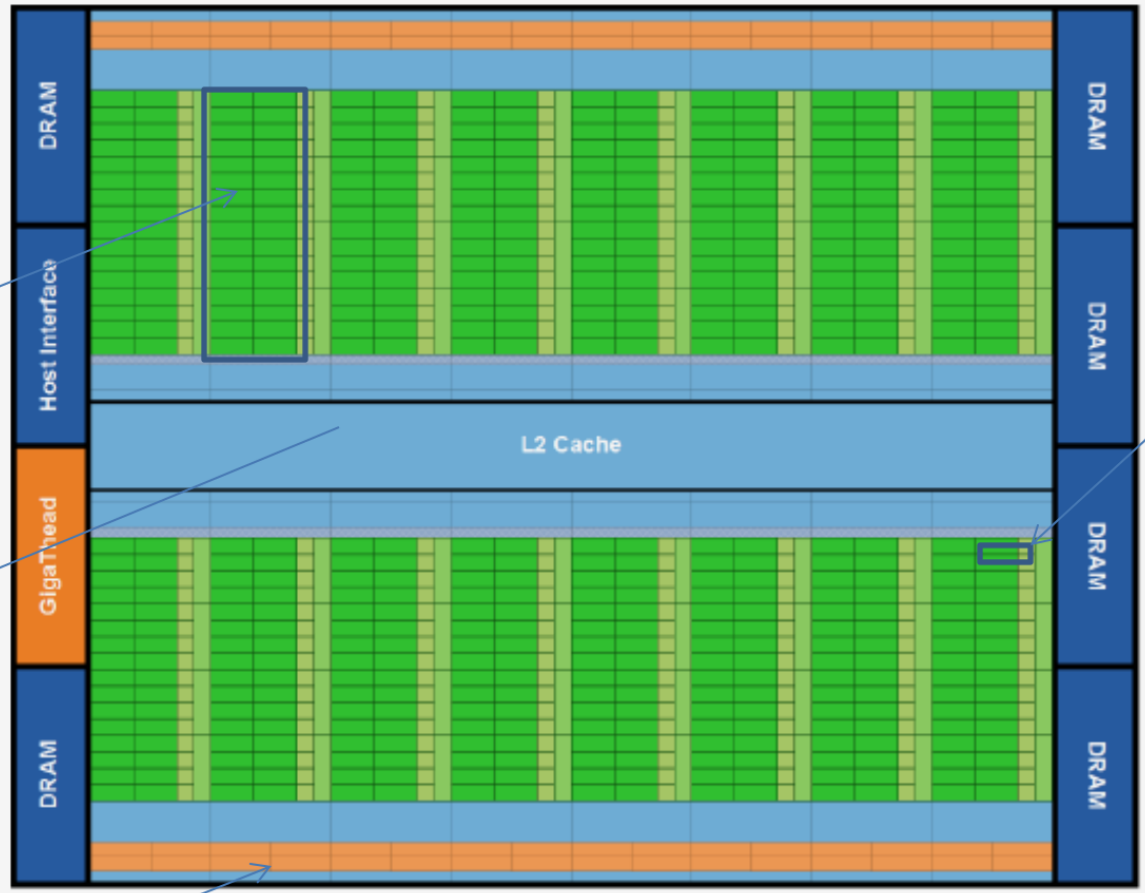
Motivations des coprocesseurs face aux CPU

- Certains codes numériques se prêtent bien au paradigme SIMD, où l'on applique un seul procédé un des nombreuses données en même temps.
- Les opération superscalaires sont limitées à 512bits sur un petit nombre de coeurs CPU.
- Il est possible de créer des coeurs de calcul avec beaucoup moins de fonctionnalités que des coeurs CPU, et qui s'exécutent en même temps par groupes (ils ne sont plus indépendants).
- Historiquement ce type d'architecture nous vient surtout des cartes graphiques.
- Des modèles et des environnements de programmation de plus en plus mûres (développés depuis 2005).
- Développer plus de puissance de calcul par noeud.

Architecture des GPU



Architecture des GPU



Multiprocesseur SM

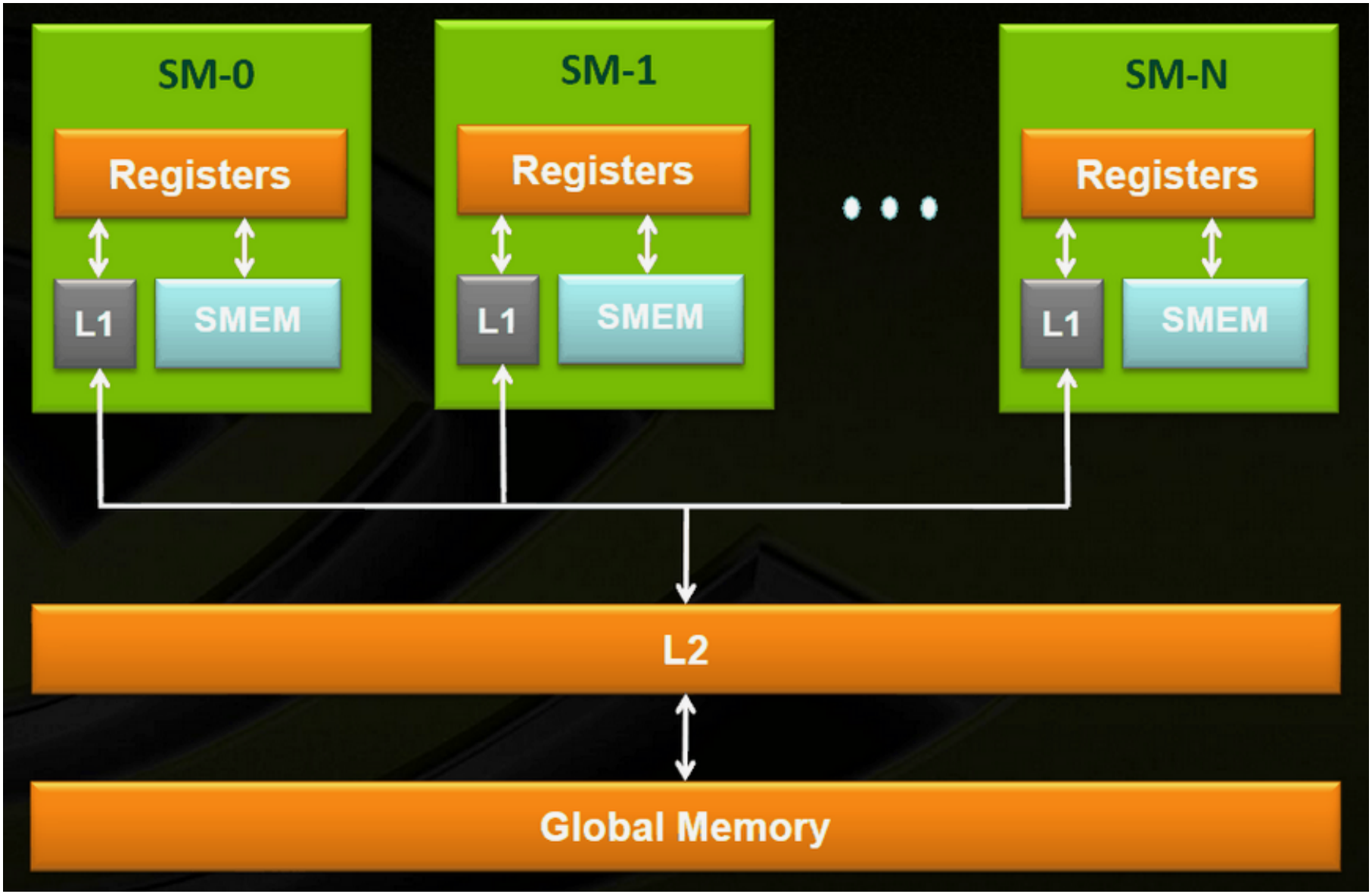
core

Cache L2 de 768 Ko

Shared memory / cache L1
64 Ko/SM

512 Unités de calcul !
16 partitions de 32

Architecture des GPU



Les cartes graphiques concurrentes (gamme pro.) :

Nvidia Tesla P100

AMD Firepro W9100

C'est la gamme dédiée au calcul en double précision, sans support graphique. Il y a un ratio 1/2 entre les unités de calcul simple et double précision.

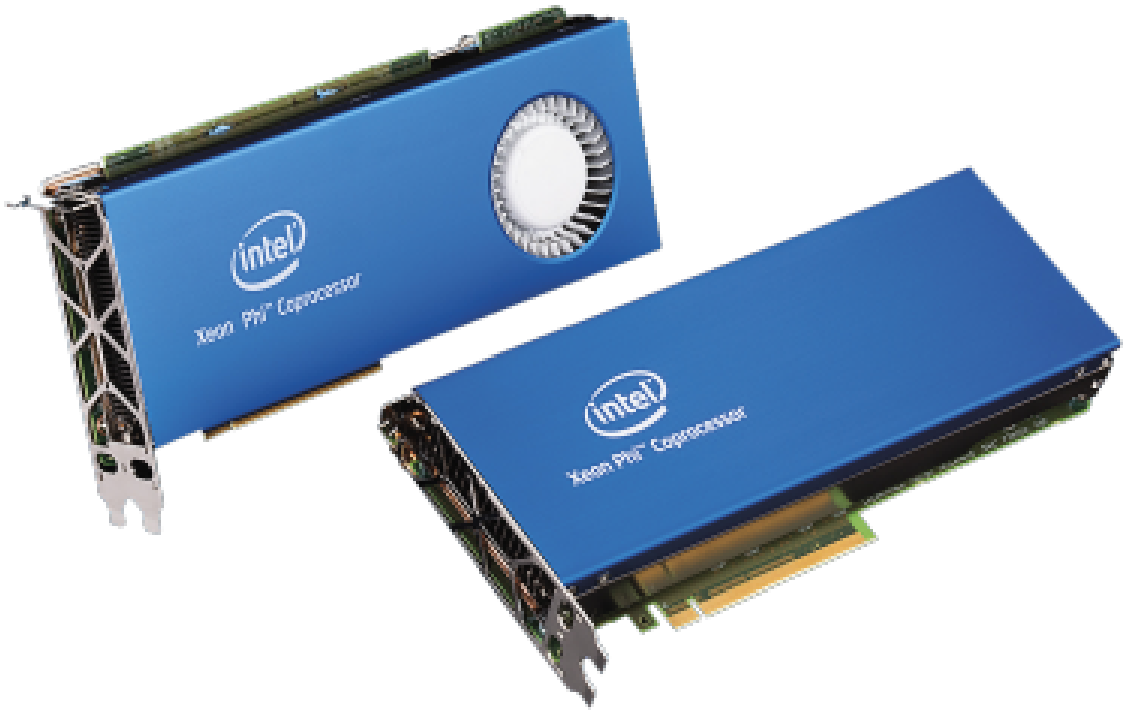
Les cartes graphiques concurrentes (gamme "gaming") :

Nvidia Titan Xp

AMD RX Vega 64

C'est la gamme dédiée au calcul en simple précision (principalement utilisé dans le rendu de scène 3d). Il y a un ratio entre 1/4 et 1/32 entre les unités de calcul simple et double précision.

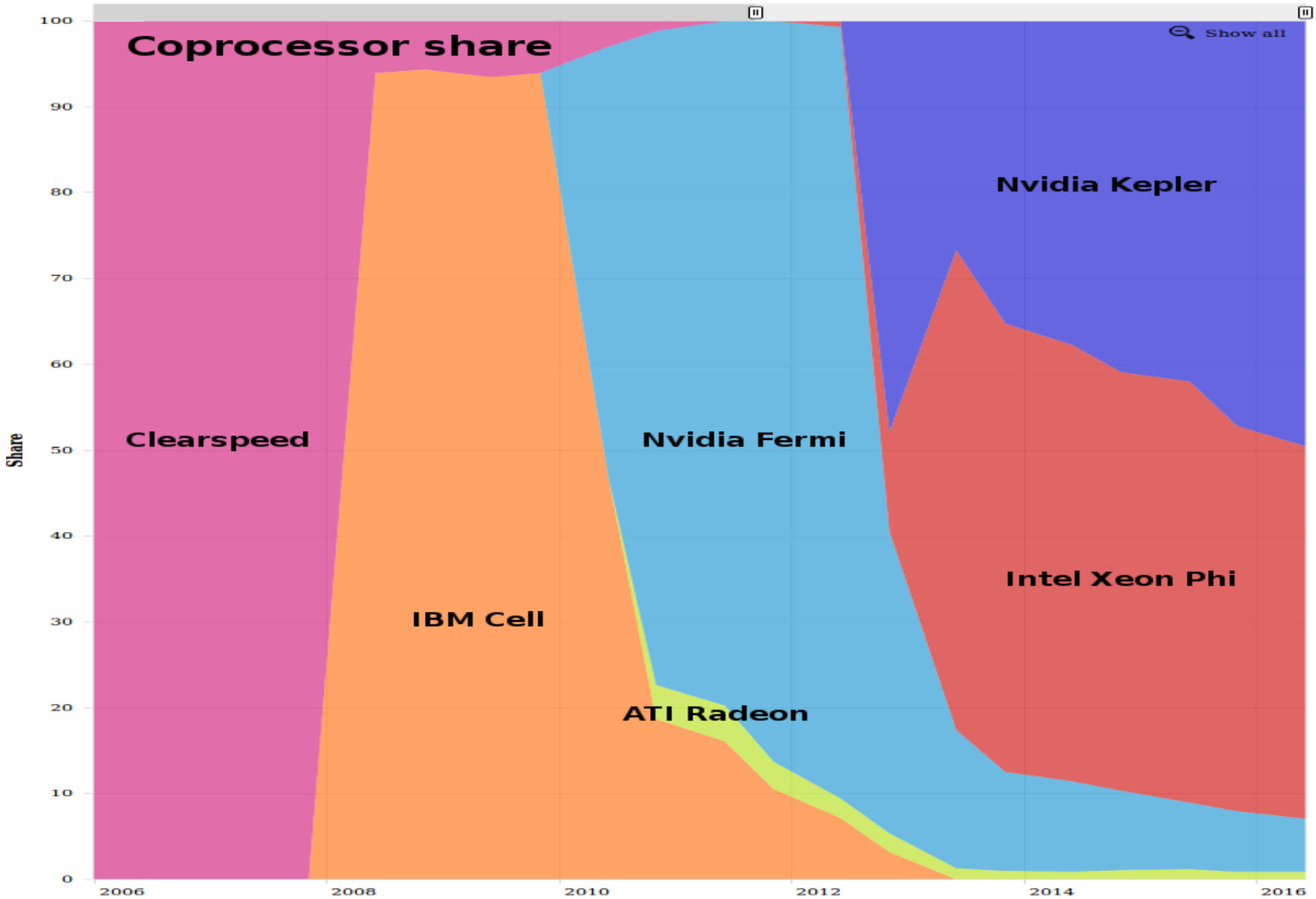
La réponse d'Intel :



Coprocesseurs MIC (Many Integrated Core) type Xeon-Phi (72 coeurs)

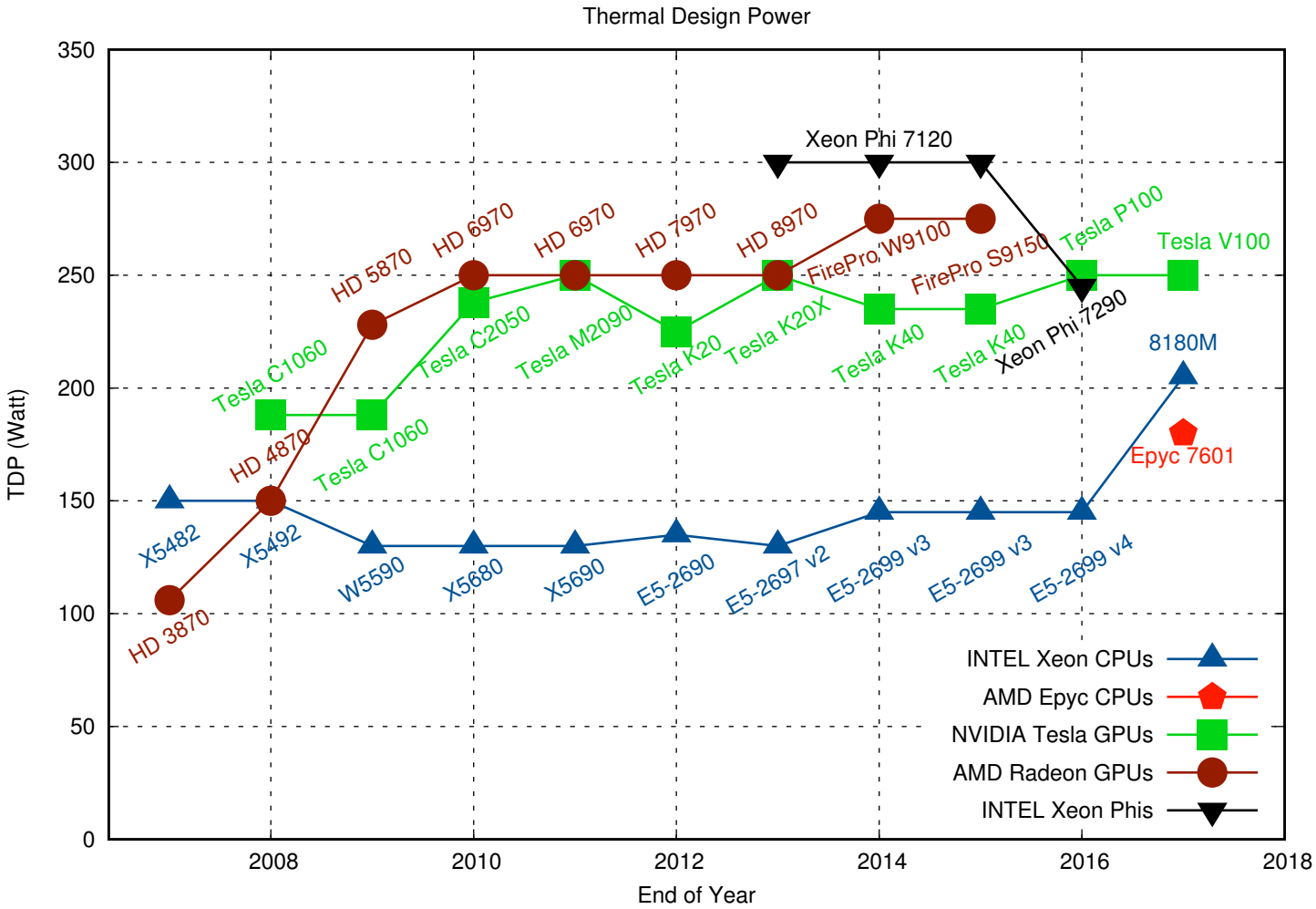
Tout comme les CPU Xeon d'Intel, ces cartes on un ratio 1/2 entre les unités de calcul simple et double précision. Ils possèdent une mémoire dédiée comme les GPU et un nombre de coeurs un peu plus élevé que les CPU Xeons traditionnels ce qui en font une architecture intermédiaire entre CPU et GPU.

Parts des systèmes avec coprocesseur dans le TOP500



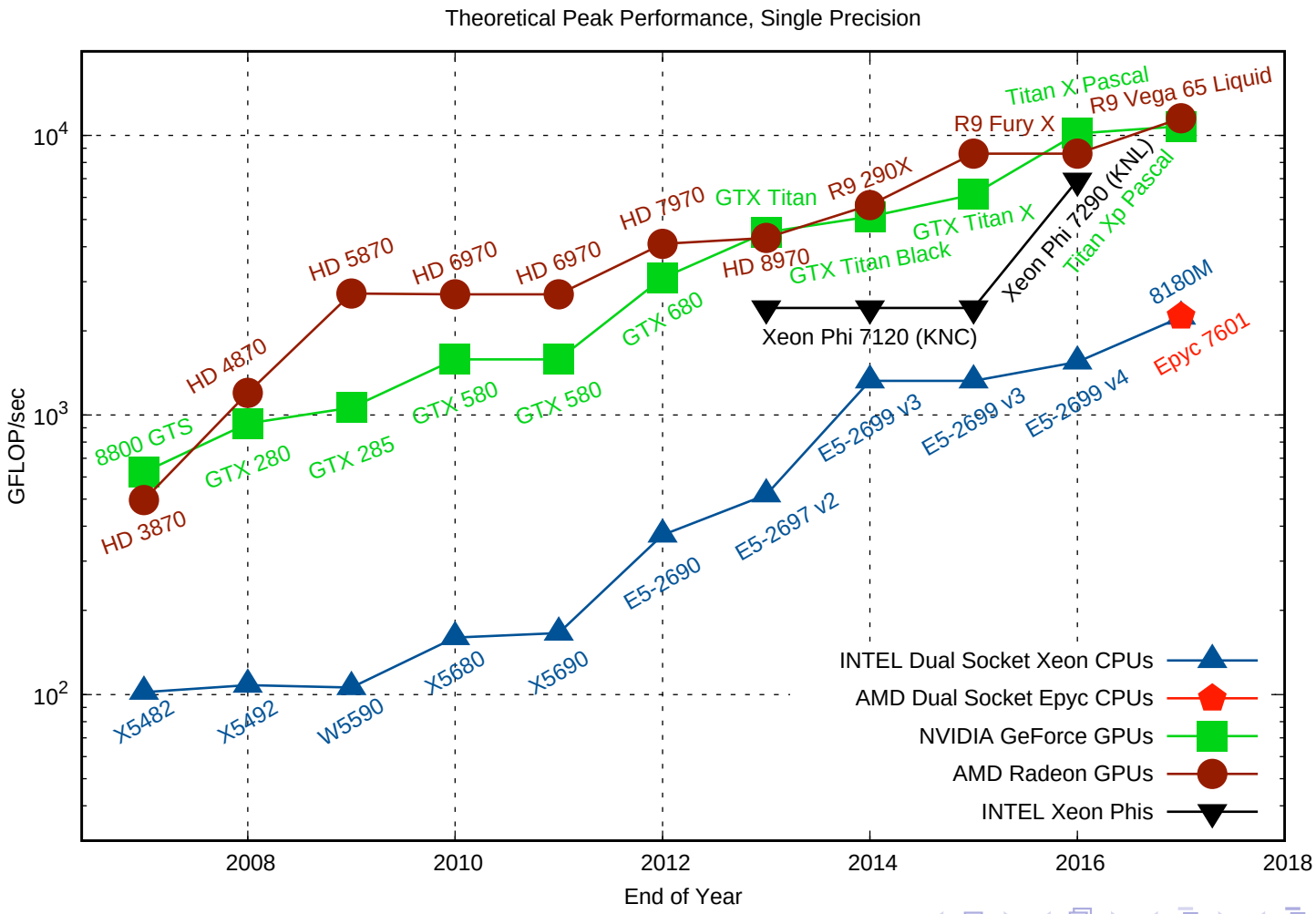
Dissipation thermique

Les processeurs sont élaborés avec une enveloppe thermique deux fois moindre par rapport aux coprocesseurs. Pour faire des comparaison équitable on considérera donc des configurations biprocesseurs.



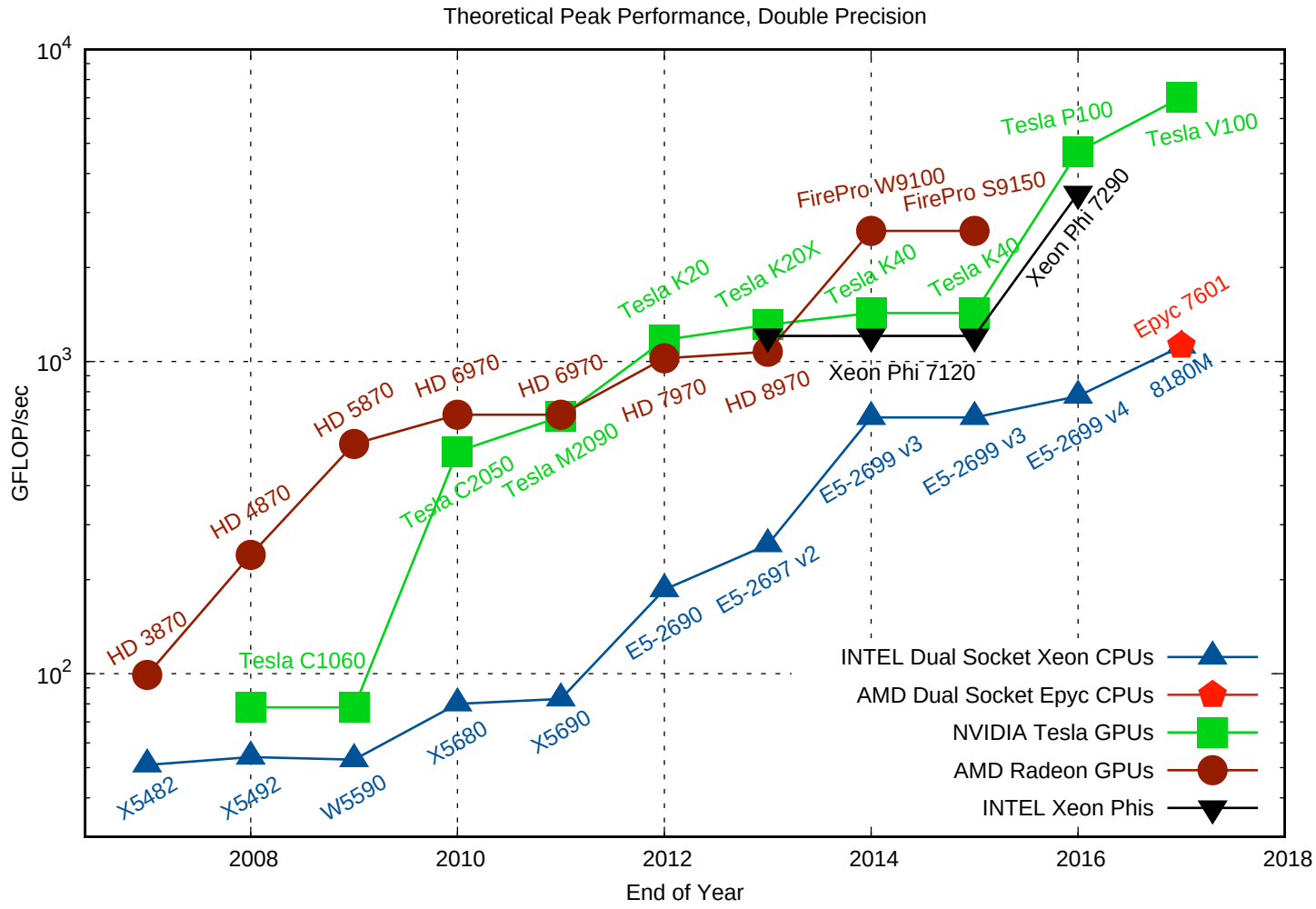
Performances brutes : Simple précision

Performances maximales théoriques mesurée en GFLOP/s (opérations à virgule flottante par seconde). L'on considère ici uniquement les gammes de GPU "gamer" dédiés aux calculs en simple précision.

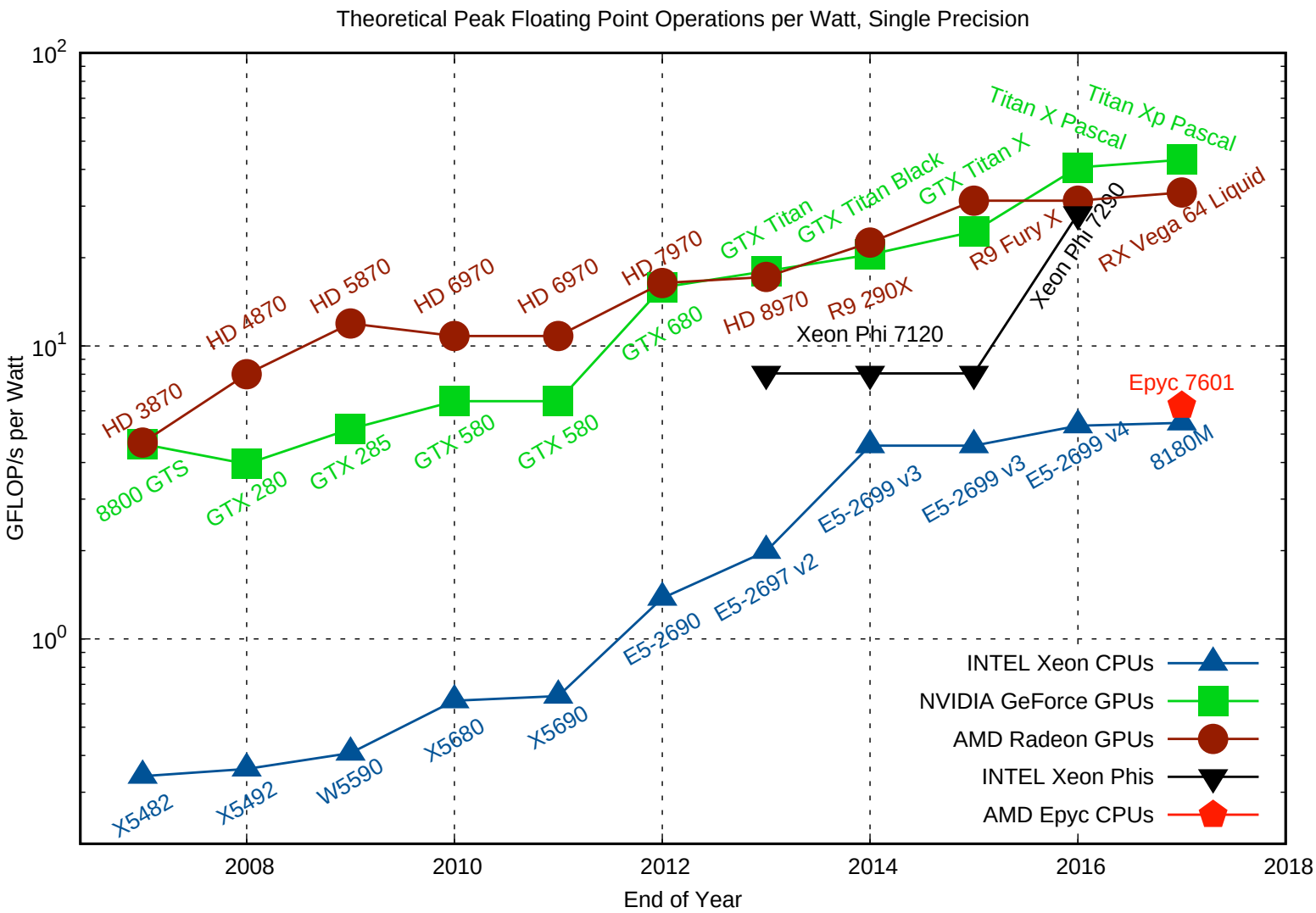


Performances brutes : Double précision

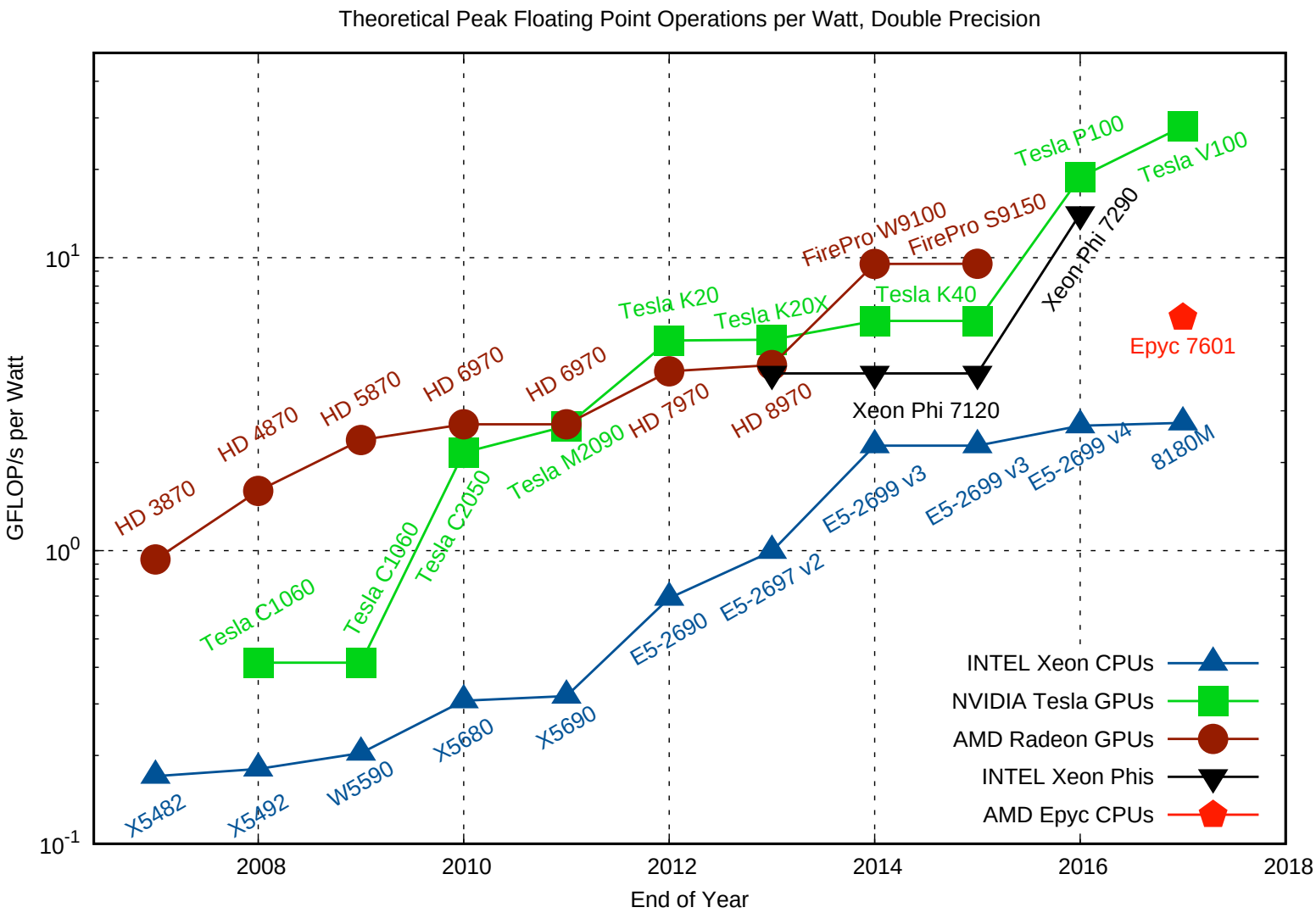
Performances maximales théoriques mesurée en GFLOP/s (opérations à virgule flottante par seconde). L'on considère ici uniquement les gammes de GPU professionnel dédiés aux calculs en double précision.



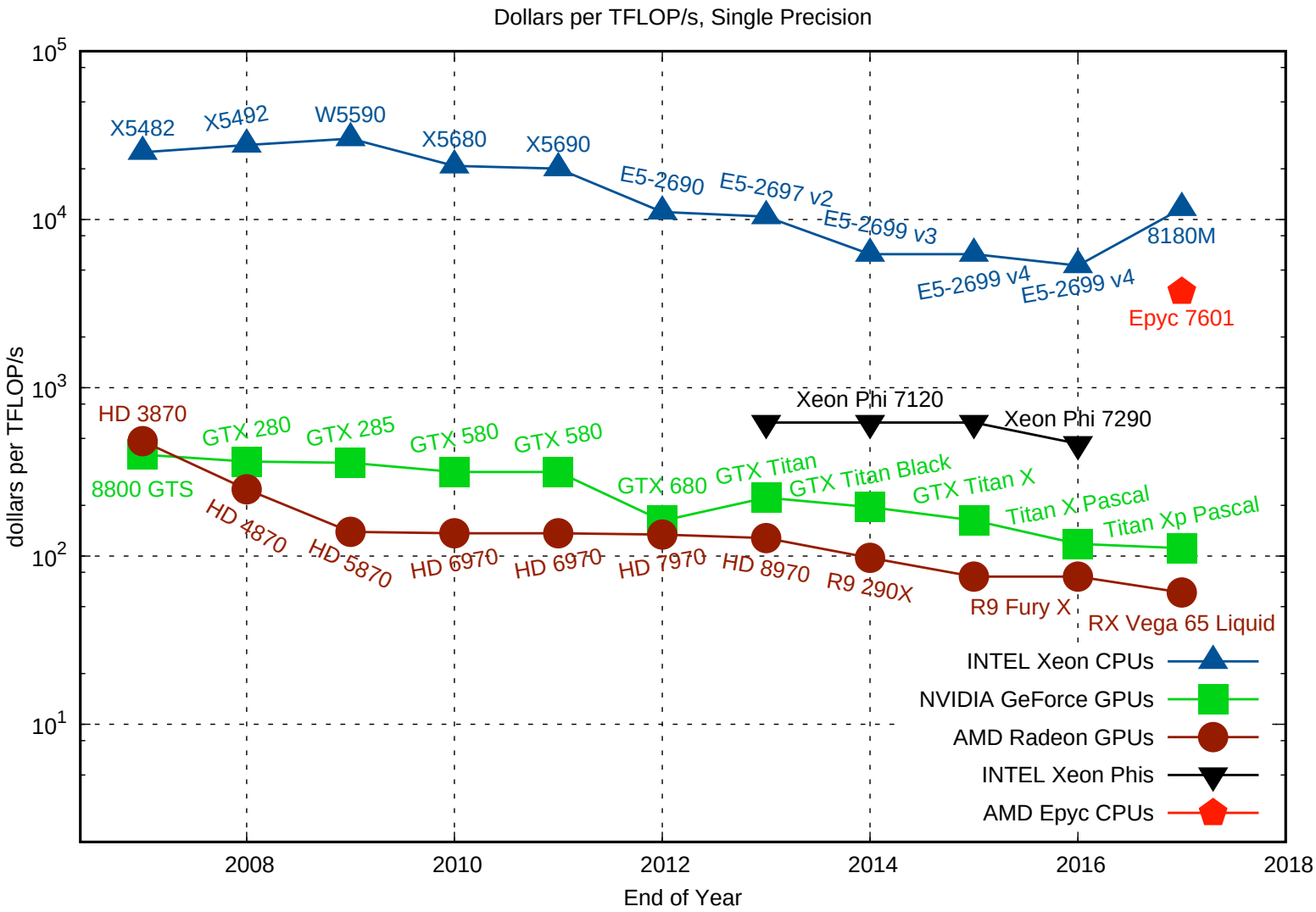
Performances brutes par Watt : Simple précision



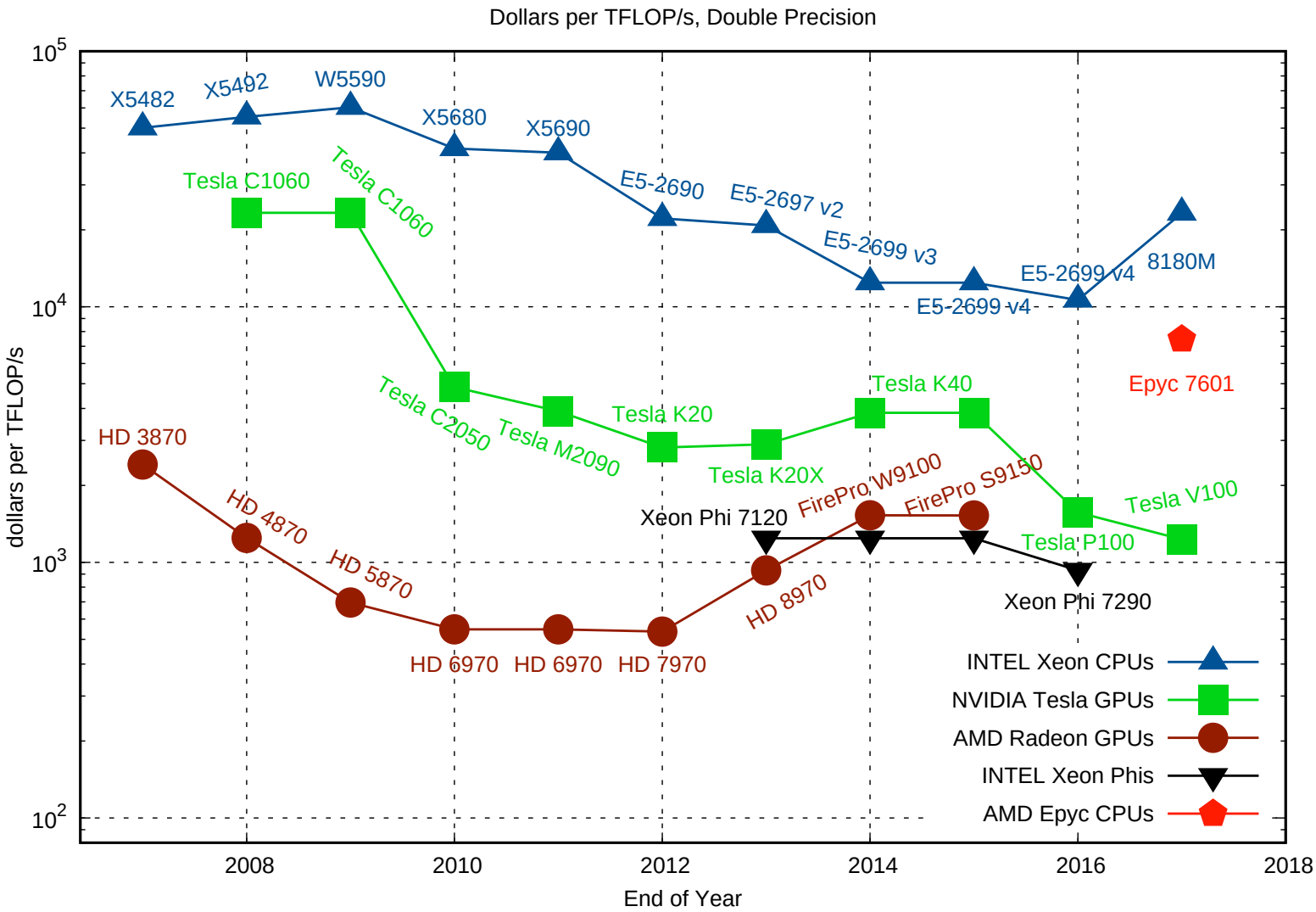
Performances brutes par Watt : Double précision



Coût d'achat par TFLOP/s délivré en simple précision

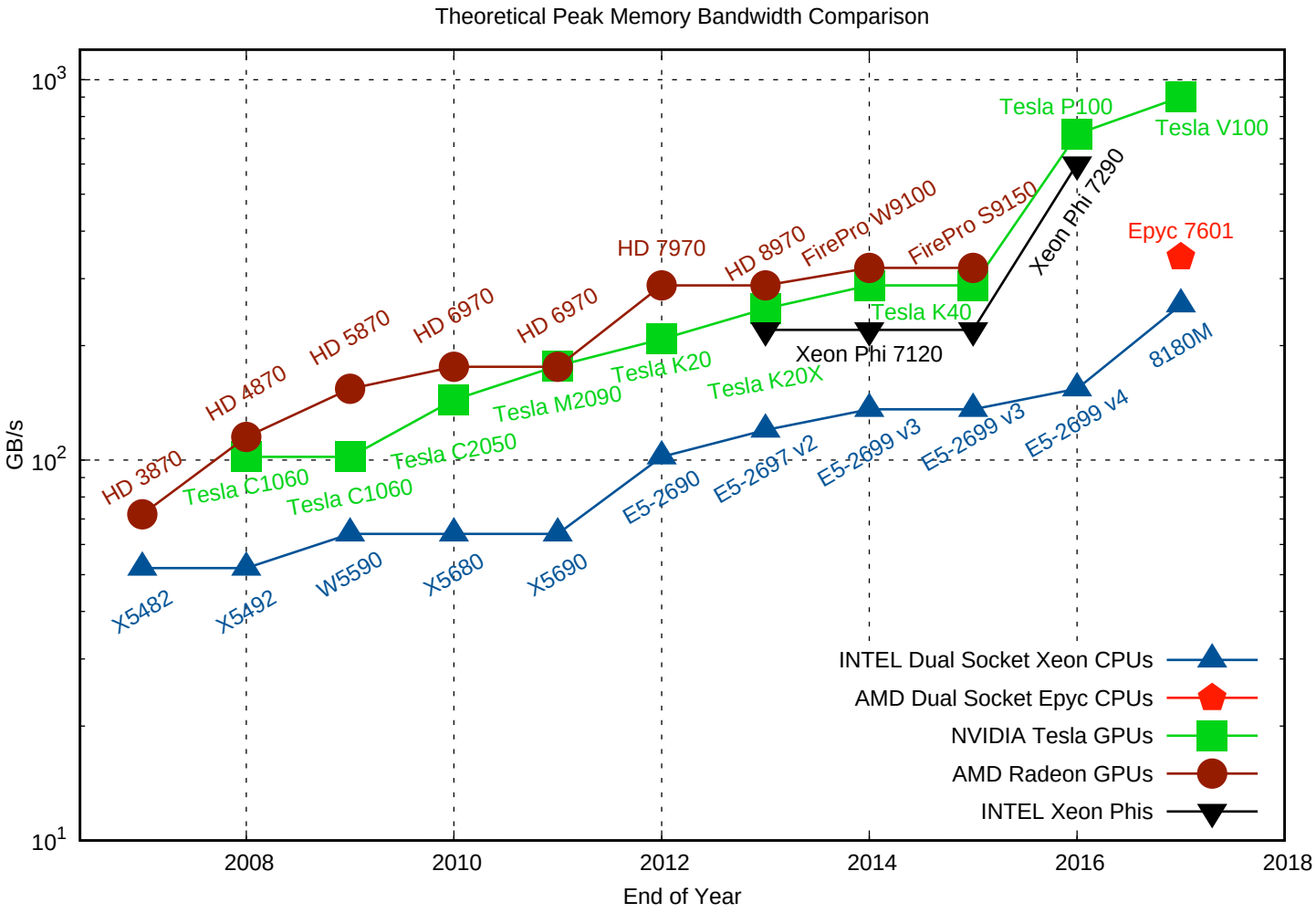


Coût d'achat par TFLOP/s délivré en double précision



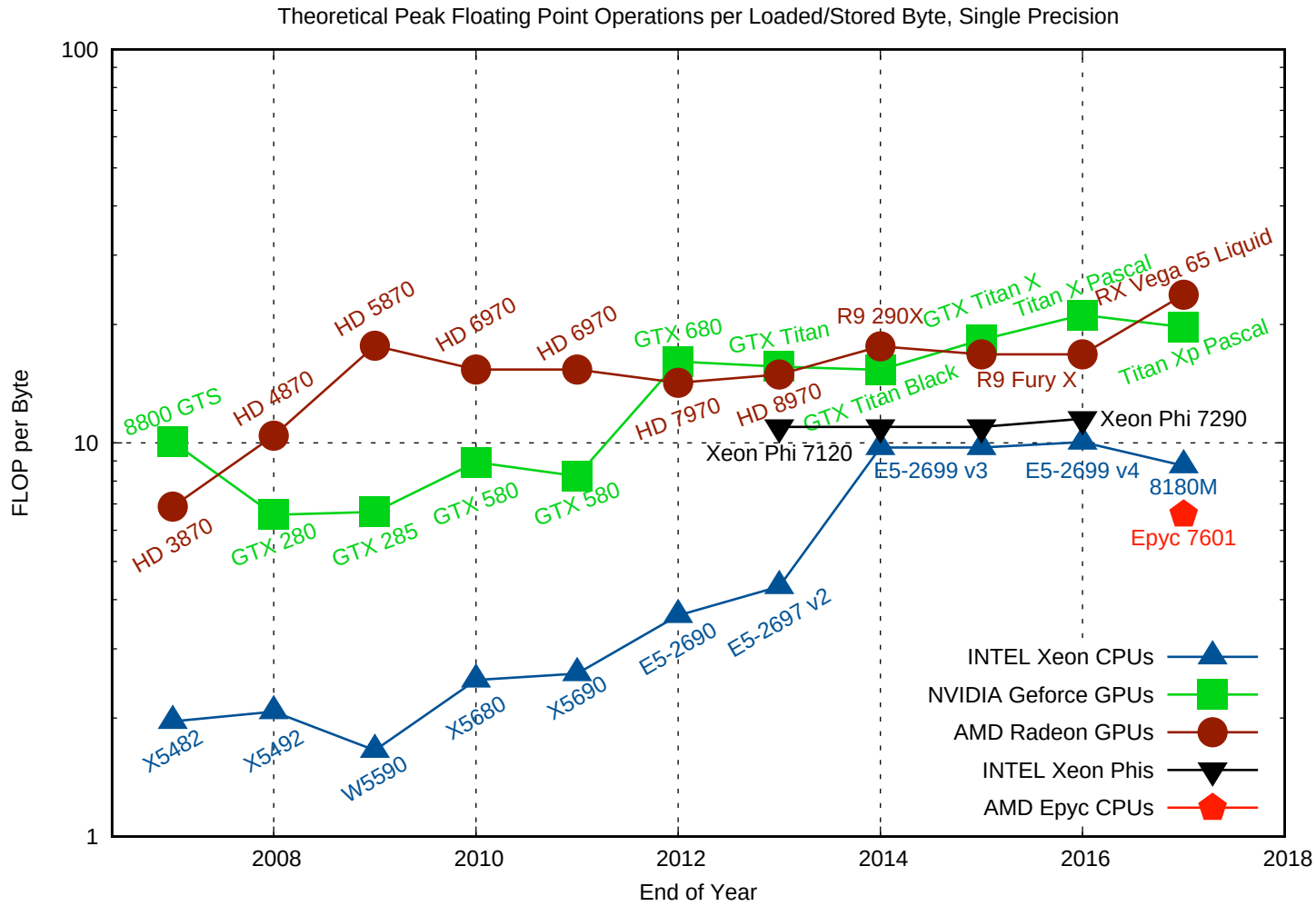
Bande passante théorique maximale

Afin d'utiliser les unités de calcul efficacement il faut que le coprocesseur soit capable de charger et stocker des données dans la mémoire globale assez rapidement.



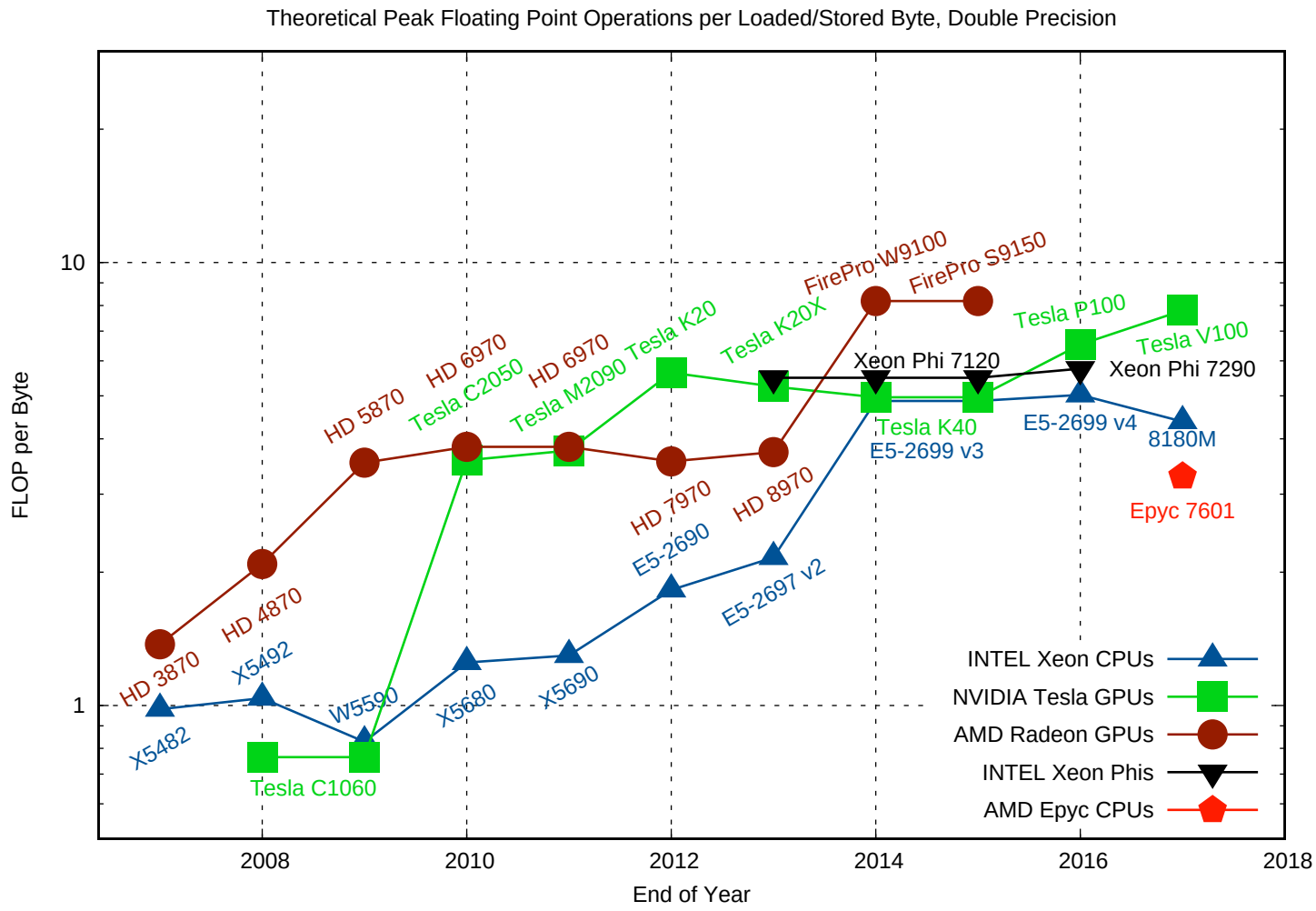
Nombre d'opérations flottantes par octet chargé (SP)

Un critère plus intéressant est le nombre minimal d'opérations flottantes à réaliser par octet chargé/stocké en mémoire afin de ne pas être limité par la bande passante mémoire. Un flottant simple précision = 4 octets.



Nombre d'opérations flottantes par octet chargé (DP)

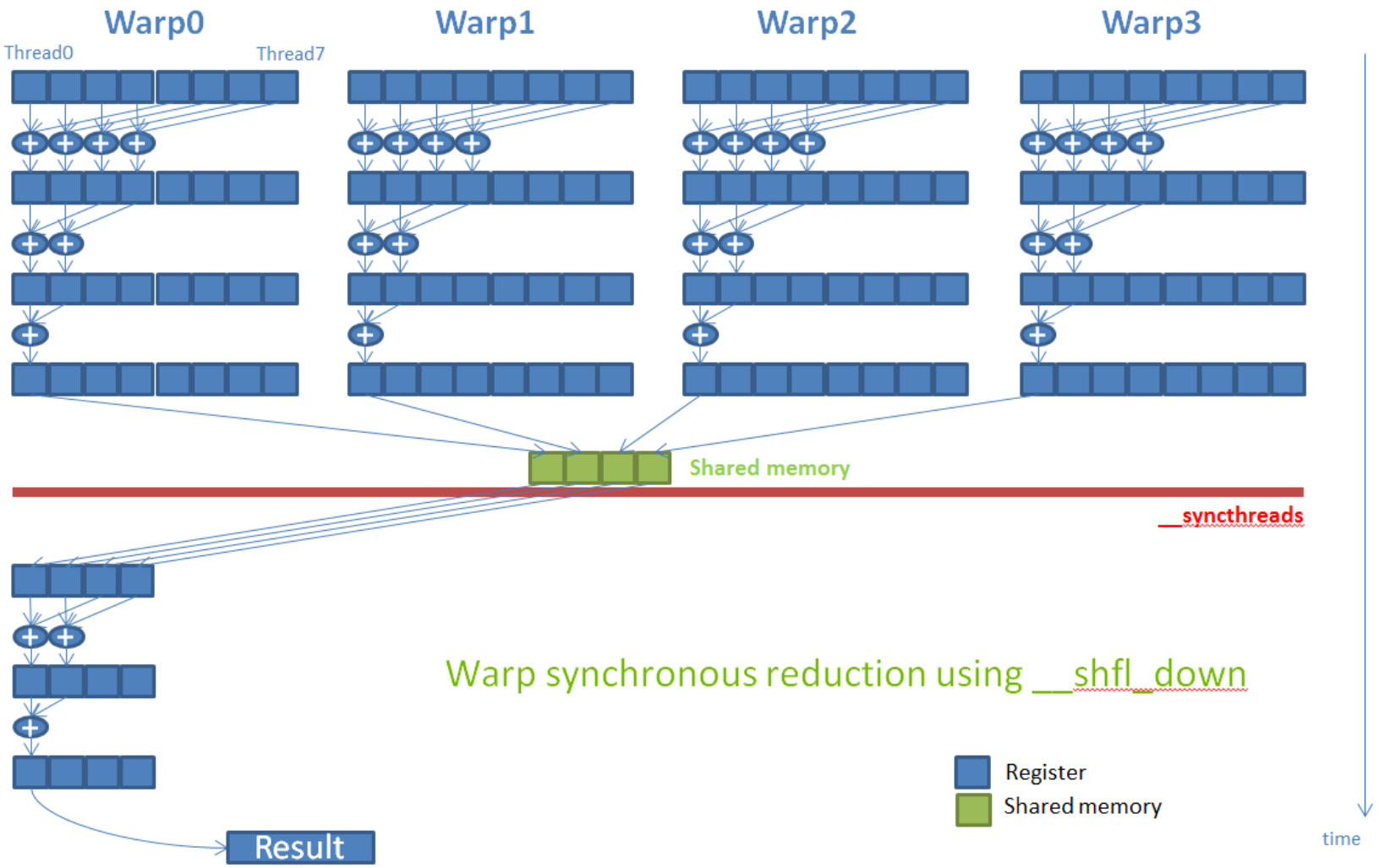
Un critère plus intéressant est le nombre minimal d'opérations flottantes à réaliser par octet chargé/stocké en mémoire afin de ne pas être limité par la bande passante mémoire. Un flottant double précision = 8 octets.



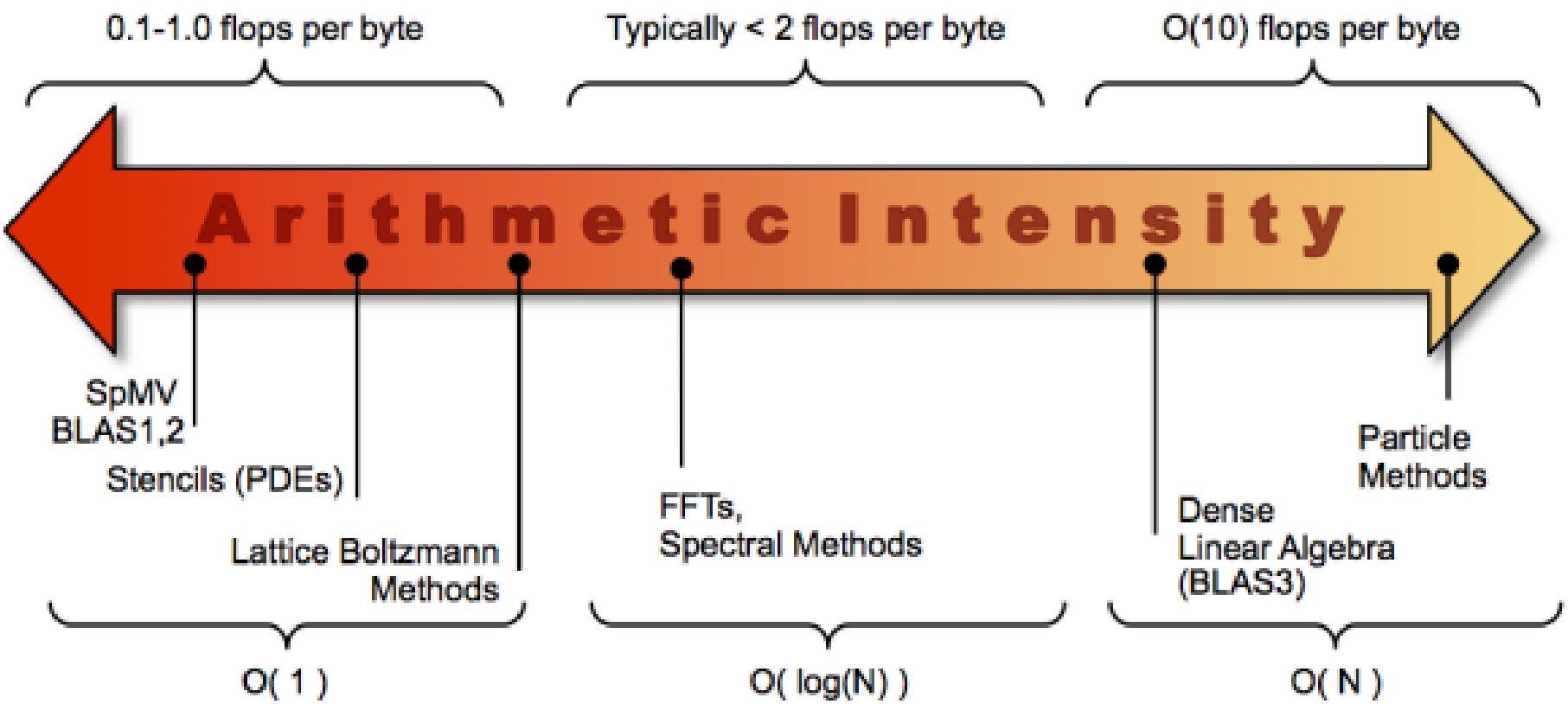
Les algorithmes qui marchent bien

Généralement : Données régulières, peu de chargement mémoire, beaucoup de calculs identiques et indépendants.

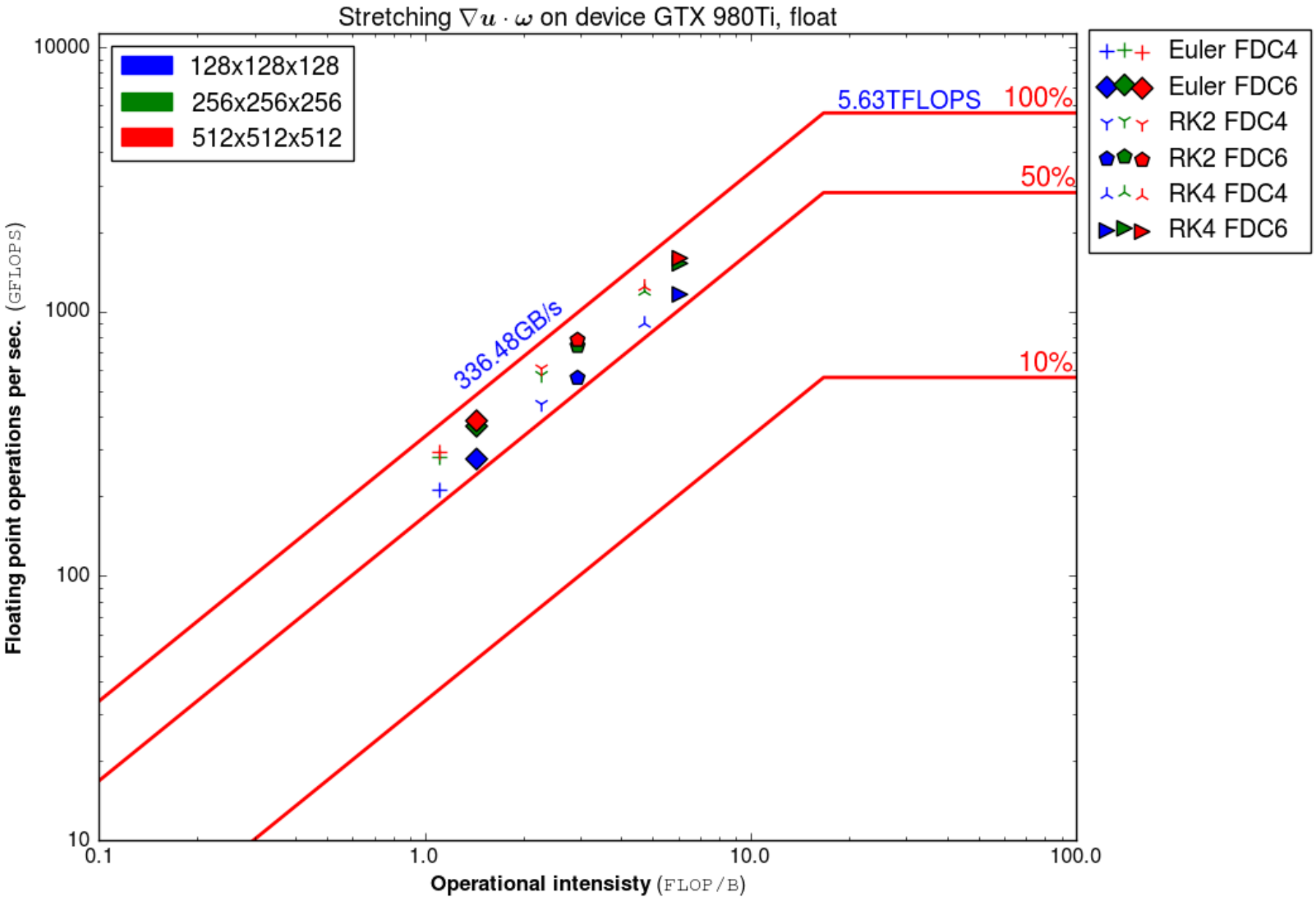
Exemple : algorithme de reduction (loi associative : max, +, ...)



Intensité arithmétique



Modèle roofline



Modèles de programmation pour coprocesseur

- OpenCL (Open Computing Language) : Permet l'exécution de code à la fois sur CPU, GPU et MIC. Cela n'est pas synonyme de portabilité des performances car il faut tuner le code pour chaque architecture.
  1. OpenCL 1.x : Basé sur un sous ensemble du C99 (2008).
  2. OpenCL 2.x : Extension sur un sous ensemble du C++14 (2013).

  C'est le standard ouvert libre de droit que chaque fabriquant de puce peut décider d'implémenter.
- CUDA (Compute Unified Device Architecture) : Equivalent propriétaire d'OpenCL pour les GPU Nvidia. Basé sur un sous ensemble du C99 et étendu à un sous ensemble du C++14 (2007). CUDA possède un écosystème d'outils et de librairies plus matures qui incluent notamment un profileur et un débogueur.
- OpenACC (Open Accelerators) : Comme OpenMP, ajout de commandes préprocesseur directement dans le code source (2012).
- OpenMP (Open Multi-Processing) : depuis sa version 4.0 (2013).

Support des modèles de programmation

	Intel		**AMD**		**Nvidia**
	CPU	MIC	CPU	GPU	GPU
OpenCL	2.1	1.2	2.0	1.2	1.2
CUDA	-	-	-	-	9.0
OpenMP	gcc/icc	gcc/icc	gcc	gcc	gcc
OpenACC	-	-	-	-	gcc/pgcc

Gains de performances sur quelques librairies de références

Gain d'un accélérateur Nvidia K40m (5500$) face à un processeur Intel E5-2697 12coeurs@2.70GHz (2700$) :

- cuBLAS vs mkIBLAS (Basic Linear Algebra Subroutine) : 6-17x
- cuSPARSE vs mkISPARSE (Sparse Matrix Library) : 5x
- cuRAND vs mkIRAND (Random Number Generation) : 70x
- cuFFT vs mkIFFT (Fast Fourier Transform) : 2-10x (10-50x sans copie)