le cnam

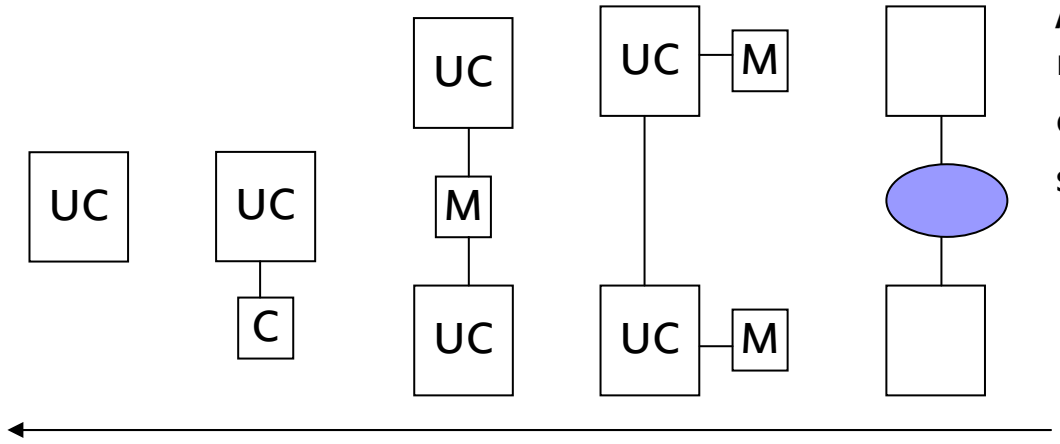
Architectures multiprocesseurs

NSY 104

Introduction

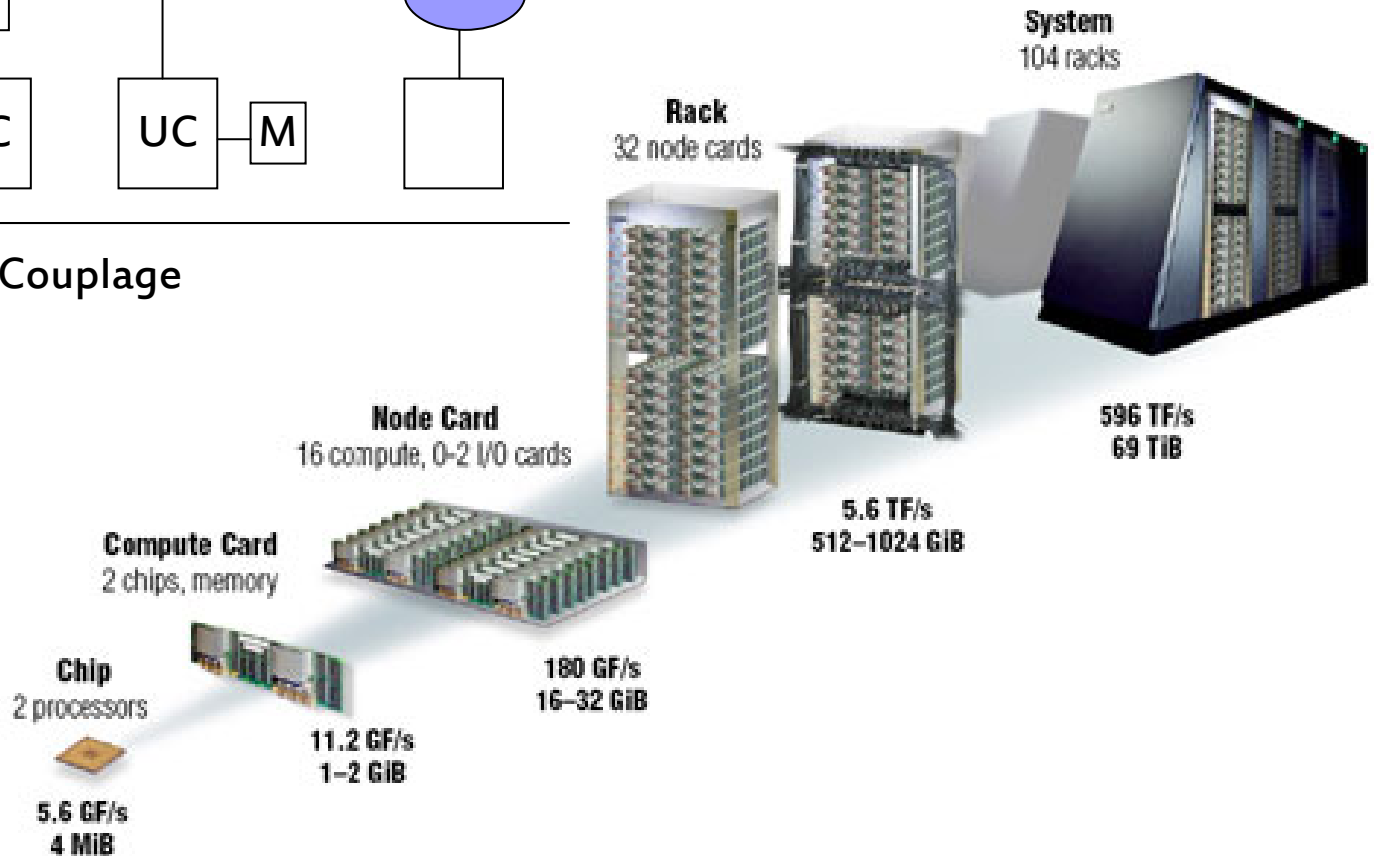
- Objectifs
  - Augmentation du nombre d'instructions traitées par seconde
  - Diminution des temps d'exécution
- Limitations
  - Technique des processeurs
  - Limites physiques de performance
- Proposition
  - Dupliquer (associer) les capacités matérielles
  - Plutôt que d'augmenter la complexité de l'unité (processeur)
    - Rapport coût-performance plus favorable

Introduction



Niveau de couplage reposant sur l'architecture mémoire et le partage de celle-ci

FLOPS :
« opérations à virgule flottante par seconde » (en anglais, *FLoating point Operations Per Second*). Le nombre de FLOPS est une mesure commune de la vitesse d'un système informatique.

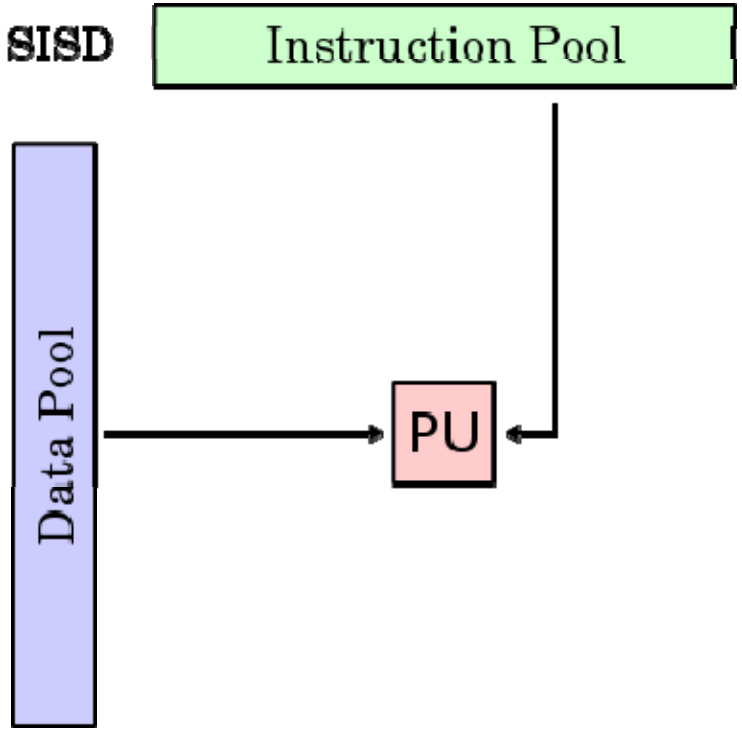


Classification des architectures

- Taxonomie de Flynn
- 4 catégories
  - SISD
    - Single Instruction Single Data
  - SIMD
    - Single Instruction Multiple Data
  - MISD
    - Multiple Instruction Single Data
  - MIMD
    - Multiple Instruction Multiple Data

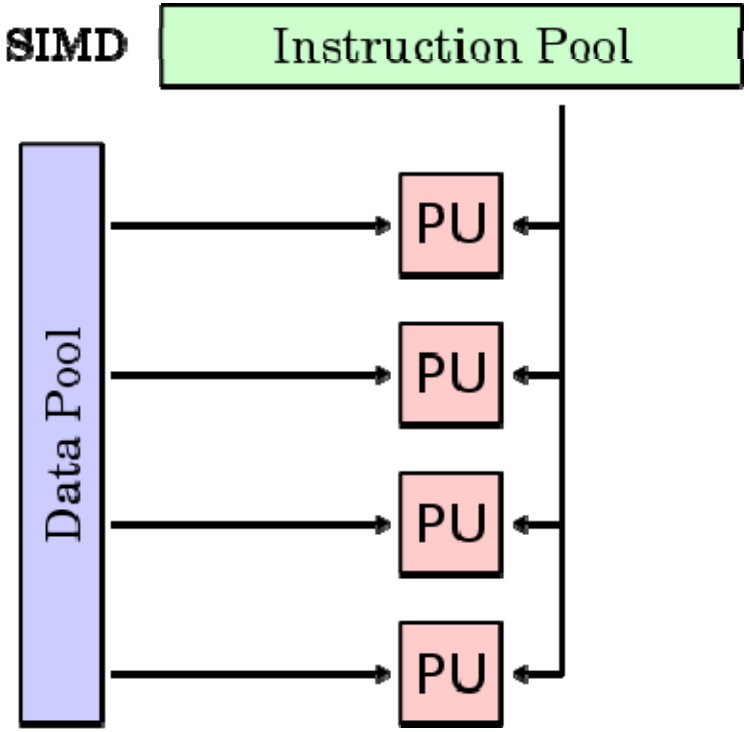
Classification des architectures

- SISD
  - Single Instruction Single Data
    - Une instruction, un flot de données
    - Ordinateur séquentiel
    - Pas de parallèlisme
      - Ordinateur monoprocesseur



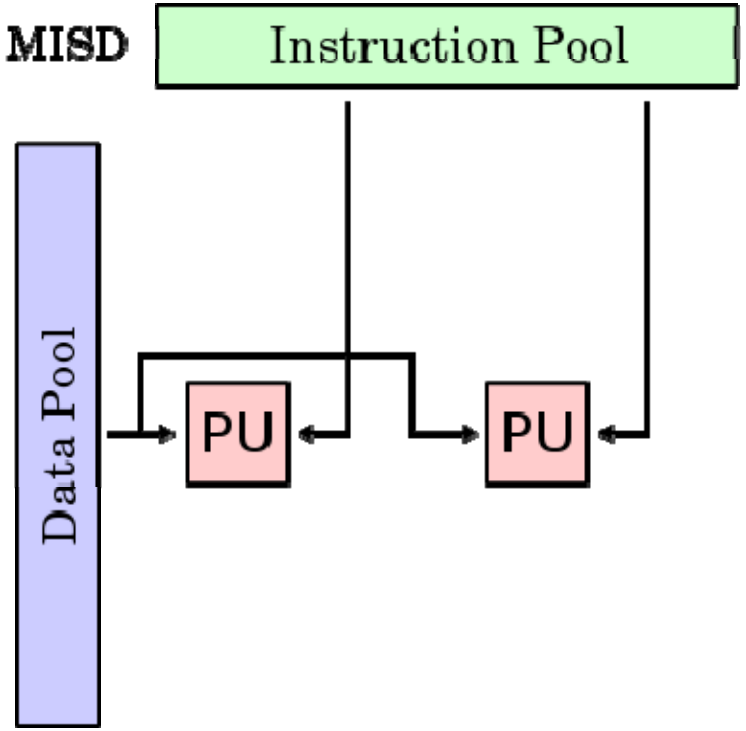
Classification des architectures

- SIMD
  - Single Instruction Multiple Data
    - Une instruction unique
    - Avec des données différentes
      - GPU
      - Processeur vectoriel
    - Synchrone
  - Exemple
    - Changement de la luminosité d'une image



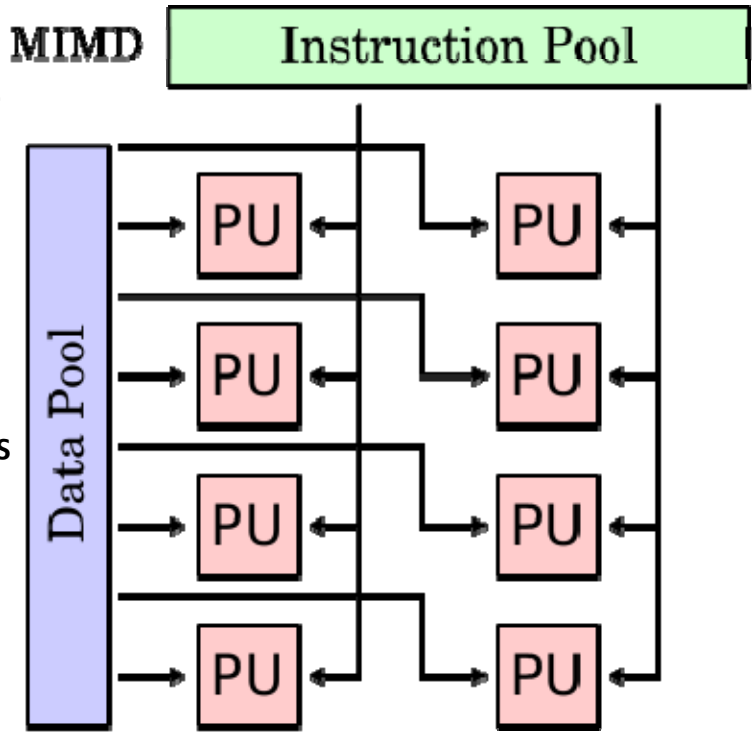
Classification des architectures

- MISD
  - Multiple Instruction Single Data
    - Instructions différentes, mêmes données
    - Rare
    - Tolérance aux pannes
      - Comparaison des résultats
      - Systèmes critiques
    - Exemple
      - Contrôleur vol navette spatiale
      - Pas (encore) de version commerciale



Classification des architectures

- MIMD
  - Multiple Instruction Multiple Data
    - Différentes instructions, différentes données
  - Deux sous groupes
    - SPMD
      - Le même programme est exécuté sur les différentes unités
      - Mais de manière indépendante
        - Asynchrone (!=SIMD)
    - MPMD
      - Différents programmes sont exécutés sur les différentes unités
      - SPU/PPU (PS3)



Classification des architectures

- Modèle simplificateur
  - Une architecture peut se trouver dans plusieurs de ces catégories

- Mais il permet de quadriller les options de conception

MIMD

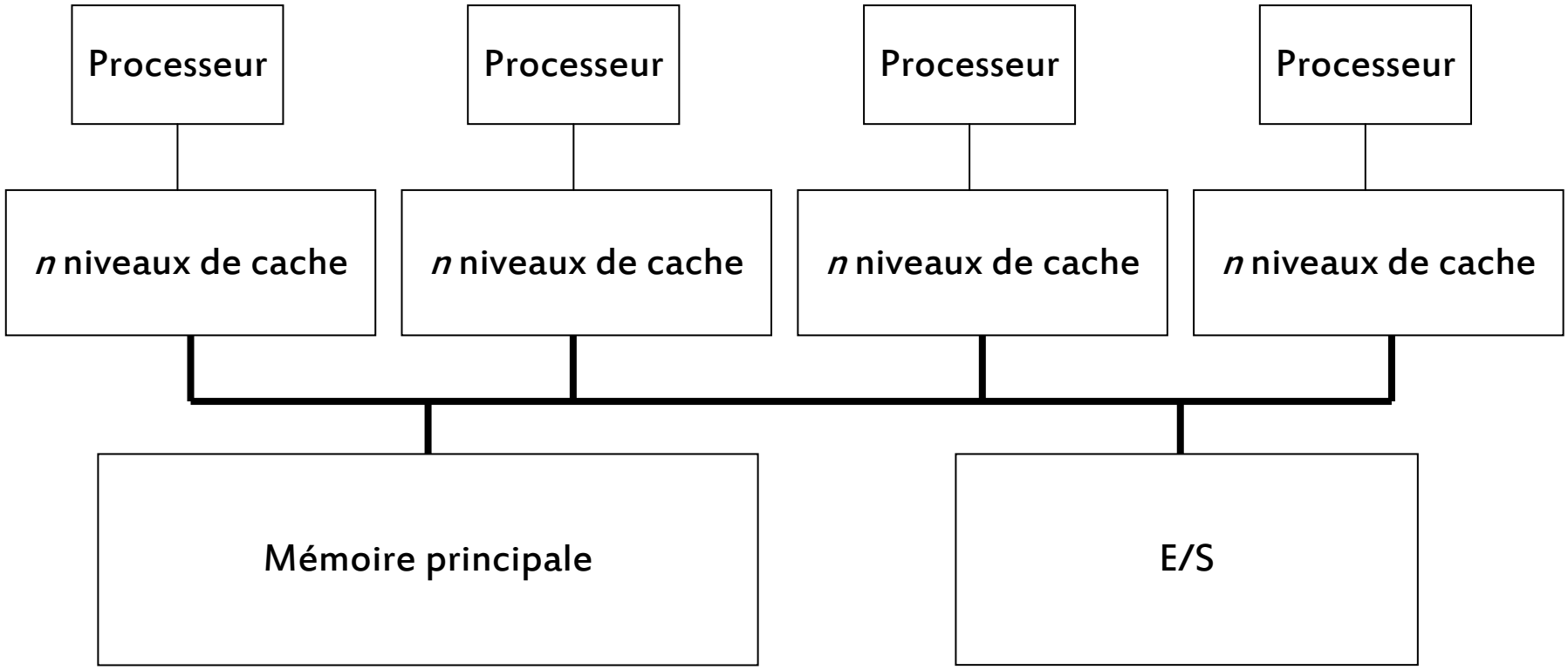
- Modèle actuel dominant
  - Flexible
    - Multiprocesseur monoprogrammé
    - Multiprocesseur multiprogrammé
    - Combinaison des deux
  - Architecture peu couteuse
    - S'appuie sur des composants standards
- Chaque processeur peut exécuter
  - Des processus (programmes) différents
  - Des sous-parties (processus légers, *threads)* d'un programme
  - Parallélisme de flot
    - *n* processeurs <-> *n* flots

Architectures mémoire

- Architectures centralisées à mémoire partagée (SMP)
  - Plusieurs processeurs
    - Avec 1 ou plusieurs caches chacun
  - Une mémoire centrale partagée
  - Un bus d'interconnexion
  - Relation symétrique entre la mémoire et les processeurs
    - Multiprocesseur symétrique à mémoire partagée
    - *Symmétric (shared-memory) multiprocessor (SMP)*
    - Accès mémoire uniforme
      - *Uniform Memory Access (UMA)*
  - Limitation du nombre de processeurs

Architectures mémoire

Multi-processeurs symétriques - SMP



Architectures mémoire

- Si on fait augmenter le nombre de processeurs
  - La mémoire devient le goulet d'étranglement
    - Problème de bande passante

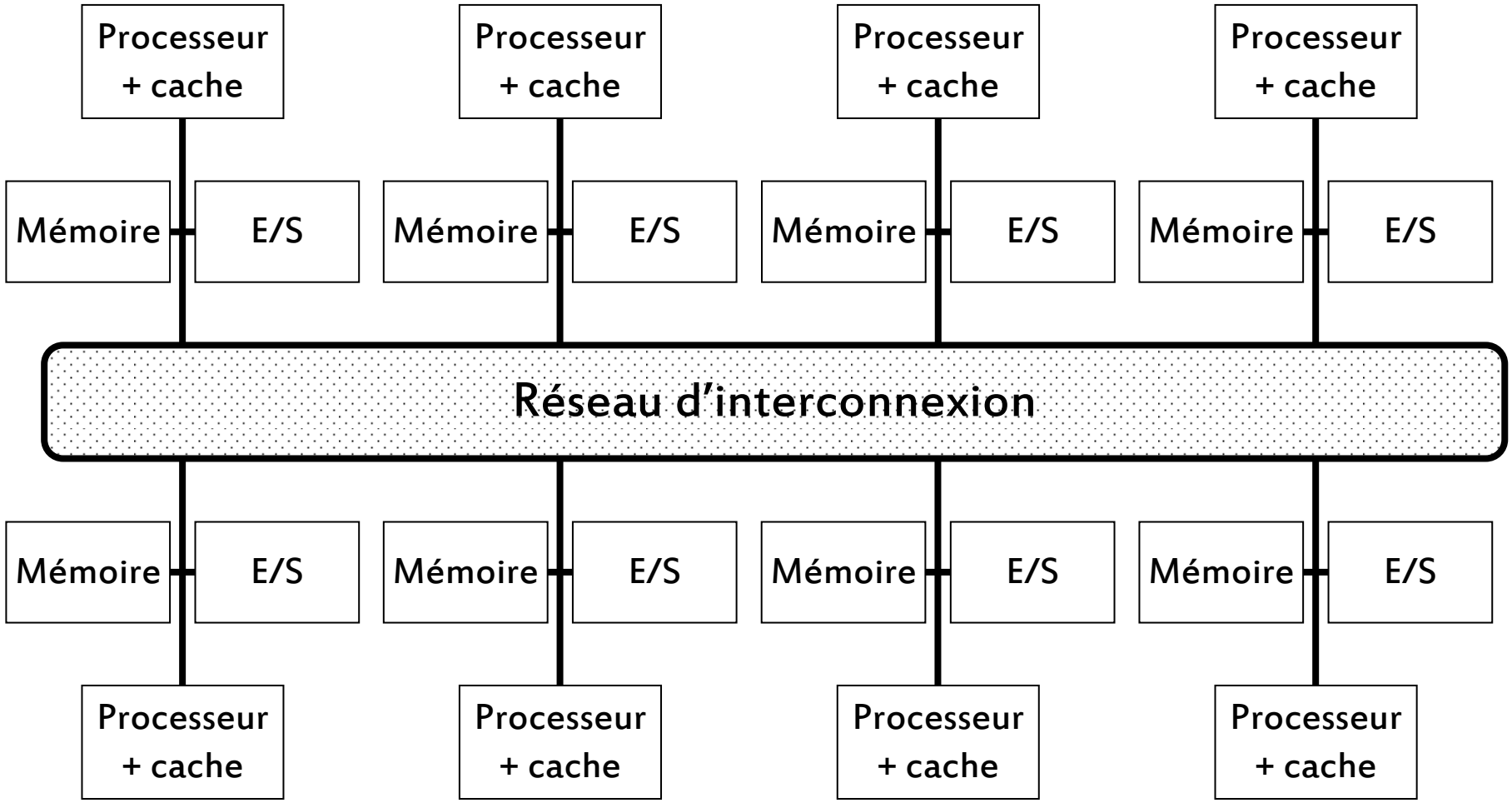
- On doit alors passer
  - d'un modèle à mémoire centralisée
  - à un modèle à mémoire distribuée

Architectures mémoire

- Architectures à mémoire distribuée
  - Mémoire physiquement distribuée
    - Meilleure bande passante
    - Plus grand nombre de processeurs
  - Bus d'interconnexion à bande passante élevée
  - Réseaux d'interconnexion
    - directs
      - commutateurs
    - indirects
      - grilles multidimensionnelles

Architectures mémoire

- Architectures à mémoire distribuée



Architectures mémoire

- Architectures à mémoire distribuée
  - Chaque nœud peut être monoprocesseur
  - ou un SMP
    - 2-8 processeurs
  - Pour les accès à la mémoire locale
    - Extension efficace de la bande passante
      - Faible coût
    - Réduction de la latence
  - Inconvénients
    - Communications entre nœuds plus complexe
    - Avec une latence plus élevée

Architectures mémoire - mémoire distribuée

- Communication entre nœuds
  - Mémoire Partagée Distribuée
    - *Distributed Shared-Memory multiprocessor (DSM)*
    - L'espace d'adressage logique est partagé entre tous les nœuds
      - Ce n'est pas une mémoire physique unique partagée
    - Tous les processeurs accèdent logiquement à toutes les mémoires
      - S'ils disposent des droits requis
    - Les multiprocesseurs DSM sont appelés NUMA
      - *Non Uniform Memory Access*
      - Par opposition aux SMP (*UMA)*
    - Protocole de communication implicite
      - adresses logiques dans les instructions de chargement/rangement

Métrique – Loi d'Ahmdal

- Gain de performance
  - Accélération
  - Gain = $Tp_{normal}/Tp_{amélioré}$
  - Gain = $Perf_{amélioré}/Perf_{sans\ amélioration}$
  - Tpa = (1-S)Tpn + S.Tpn/Ac
    - Tpn = Tps d'exec système origine
    - Tpa = Tps d'exec système amélioré
    - S = fraction du temps concerné par l'amélioration
    - Ac = accélération obtenue par l'amélioration

Métrique – CPI

- Cycle d'horloges CPU pour un programme/Nombre d'instructions

Cohérence des mémoires - SMP

- Dans les SMP, les caches stockent
  - des données privées
    - même comportement qu'en architecture monoprocesseur
  - des données partagées
    - Peuvent exister sur des caches différents
      - Migration
      - Duplication
    - Cela pose des problèmes de cohérence de cache

Temps	Event	Cache A	Cache B	Mémoire
0				1
1	CPU_A lit X	1		1
2	CPU_B lit X	1	1	1
3	CPU_A range 0 dans X	0	1	0

Cohérence des mémoires - SMP

- Suivre l'état de tous les blocs partagés

- Deux méthodes
  - Le répertoire
    - On conserve dans un endroit unique l'état de partage d'un bloc de mémoire physique.
  - L'espionnage
    - Surveillance du bus de mémoire partagée
    - Pour déterminer s'ils ont, ou non, le bloc demandé.

Cohérence des mémoires - SMP

- Deux mécanismes sont possibles pour assurer la cohérence
  - L'invalidation d'écriture
    - Assure l'accès exclusif d'une donnée à un processeur avant une écriture
    - Invalide toutes les autres copies sur une écriture

Temps	Event	Bus	Cache A	Cache B	Mémoire
0					0
1	CPU_A lit X	Miss	0		0
2	CPU_B lit X	Miss	0	0	0
3	CPU_A écrit 1 dans X	Invalid.	1		0
4	CPU_B lit X	Miss	1	1	1

  - Quand le second miss de B intervient
  - A détecte la demande, répond sur le bus et annule la réponse de la mémoire
  - Les valeurs sont alors mises à jour sur le cache de B et en mémoire

Cohérence des mémoires - SMP

- Deux mécanismes sont adoptables pour assurer la cohérence
  - La diffusion d'écriture
    - Mise à jour systématique de toutes les copies sur une écriture

Temps	Event	Bus	Cache A	Cache B	Mémoire
0					0
1	CPU_A lit X	Miss	0		0
2	CPU_B lit X	Miss	0	0	0
3	CPU_A écrit 1 dans X	Diffusion	1	1	1
4	CPU_B lit X		1	1	1

Cohérence des mémoires – Mem. Distr.

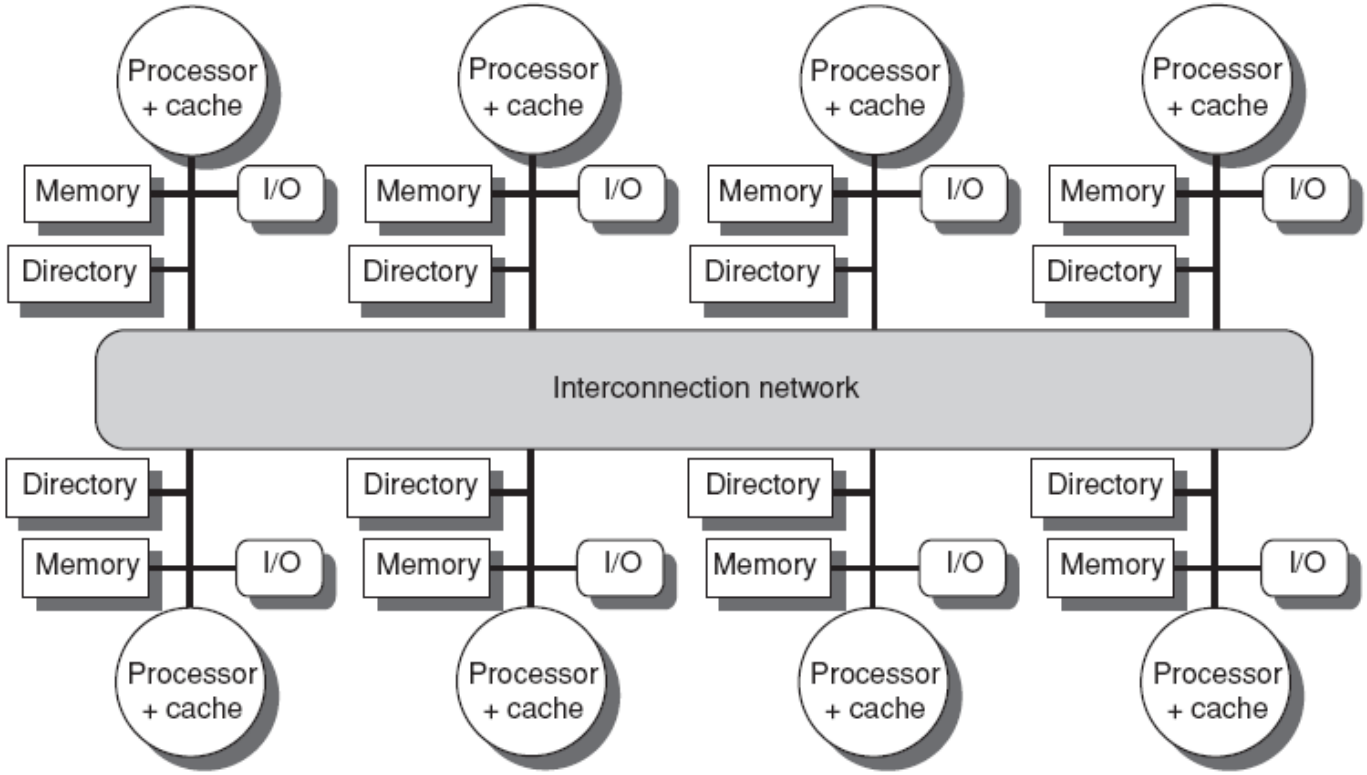
- Dans les architectures à mémoire distribuée
  - Soit un mécanisme de cohérence de cache est inclus
  - Soit il est exclu
    - Solution la plus aisée
    - Dans ce cas, seules les données privées sont dans les caches
    - Les données partagées s'obtiennent via des messages
      - Et ne sont pas « cachables »
      - Copie possible via le logiciel, qui contrôle alors la cohérence
    - Inconvénients
      - Cette gestion logicielle, via la compilation, n'est pas assez précise
      - Et ne permet pas d'être efficace
      - Perte des bénéfices de la localité spatiale
        - Car on transmet des mots, non plus des blocs, via les messages
      - Pour cette même raison, les latences sont augmentées.

Cohérence des mémoires – Mem. Distr.

- S'il est inclus
  - Ce peut être via l'espionnage
    - Très couteux en matériel
      - Communication avec tous les caches indispensable
    - Peu compatible avec les extension d'architecture
  - Le répertoire est une autre solution
    - Mieux adaptée
    - Mais qui peut devenir un goulet d'étranglement
      - si grand nombre de processeurs (> 200)
    - Nécessité d'envisager un répertoire distribué

Cohérence des mémoires – Mem. Distr.

- Répertoire distribué
  - L'état d'un bloc est stocké
    - dans le répertoire
    - à un seul endroit dans le réseau



Cohérence des mémoires – Mem. Distr.

- Répertoire distribué
  - Objectifs
    - Gérer les échecs de lecture
    - Gérer l'écriture pour un bloc partagé non modifié
  - 3 états possibles pour chaque bloc
    - Partagé
      - Le bloc est présent dans le cache d'un ou plusieurs processeurs
      - La valeur en mémoire, comme dans les caches, est à jour
    - Non caché
      - Aucun processeur n'a de copie du bloc
    - Exclusif
      - Un processeur à une copie du bloc et a écrit dans le bloc
      - La copie mémoire est périmée
      - Le processeur est le *propriétaire* du bloc

Cohérence des mémoires – Mem. Distr.

- Répertoire distribué
  - Connaissance des processeurs qui ont une copie du bloc, lorsqu'il est partagé
    - Pour l'invalidation sur écriture
    - Utilisation d'un vecteur de bits pour chaque bloc
      - Un bit correspondant à un processeur
      - Indique si un processeur possède une copie du bloc
    - Lorsqu'un bloc est exclusif
      - ce vecteur sert à en connaître le propriétaire

Cohérence des mémoires – Mem. Distr.

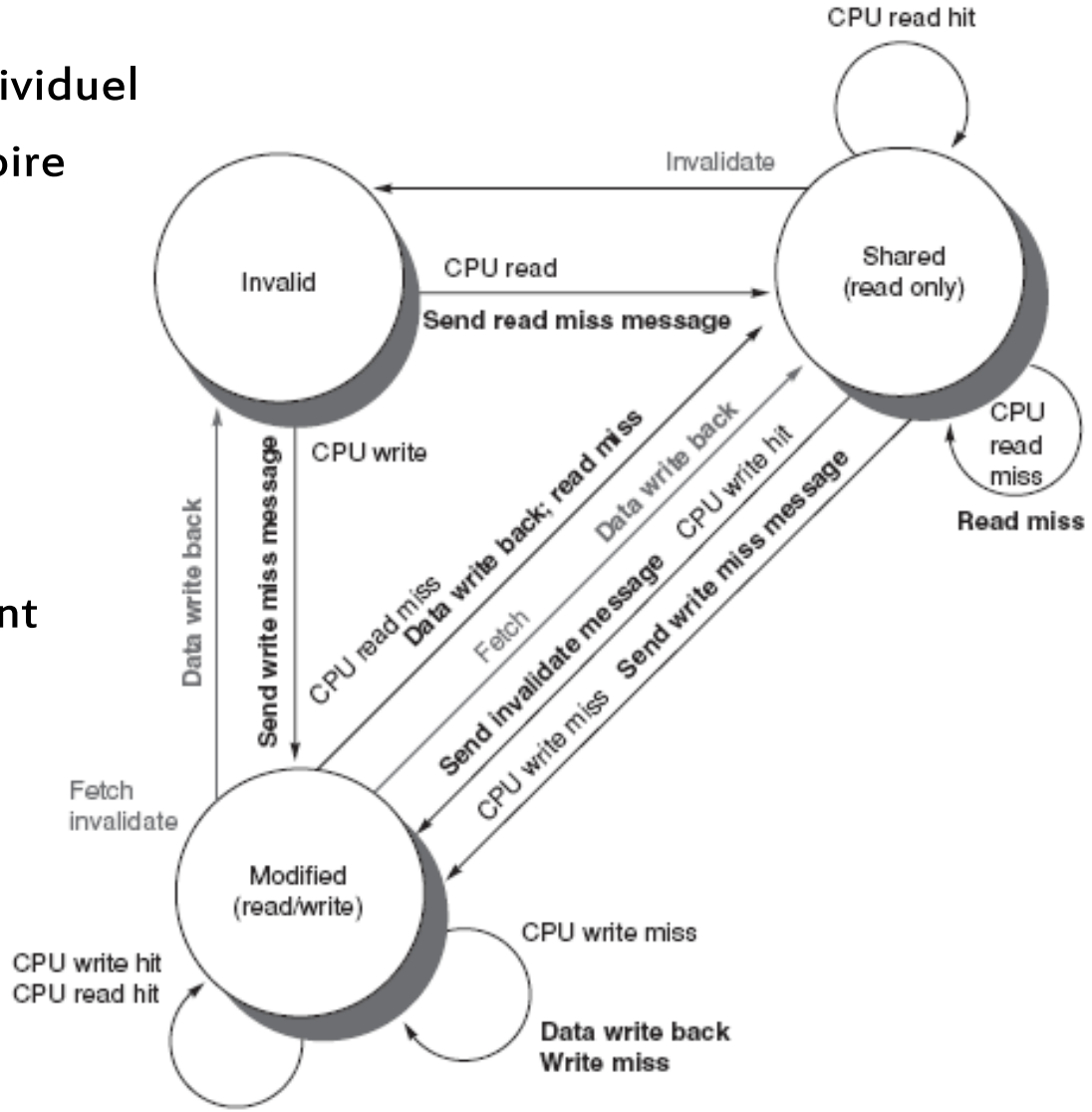
- Messages possibles entre nœuds pour le maintien de la cohérence

Message type	Source	Destination	Message contents	Function of this message
Read miss	local cache	home directory	P, A	Processor P has a read miss at address A; request data and make P a read sharer.
Write miss	local cache	home directory	P, A	Processor P has a write miss at address A; request data and make P the exclusive owner.
Invalidate	local cache	home directory	A	Request to send invalidates to all remote caches that are caching the block at address A.
Invalidate	home directory	remote cache	A	Invalidate a shared copy of data at address A.
Fetch	home directory	remote cache	A	Fetch the block at address A and send it to its home directory; change the state of A in the remote cache to shared.
Fetch/invalidate	home directory	remote cache	A	Fetch the block at address A and send it to its home directory; invalidate the block in the cache.
Data value reply	home directory	local cache	D	Return a data value from the home memory.
Data write back	remote cache	home directory	A, D	Write back a data value for address A.

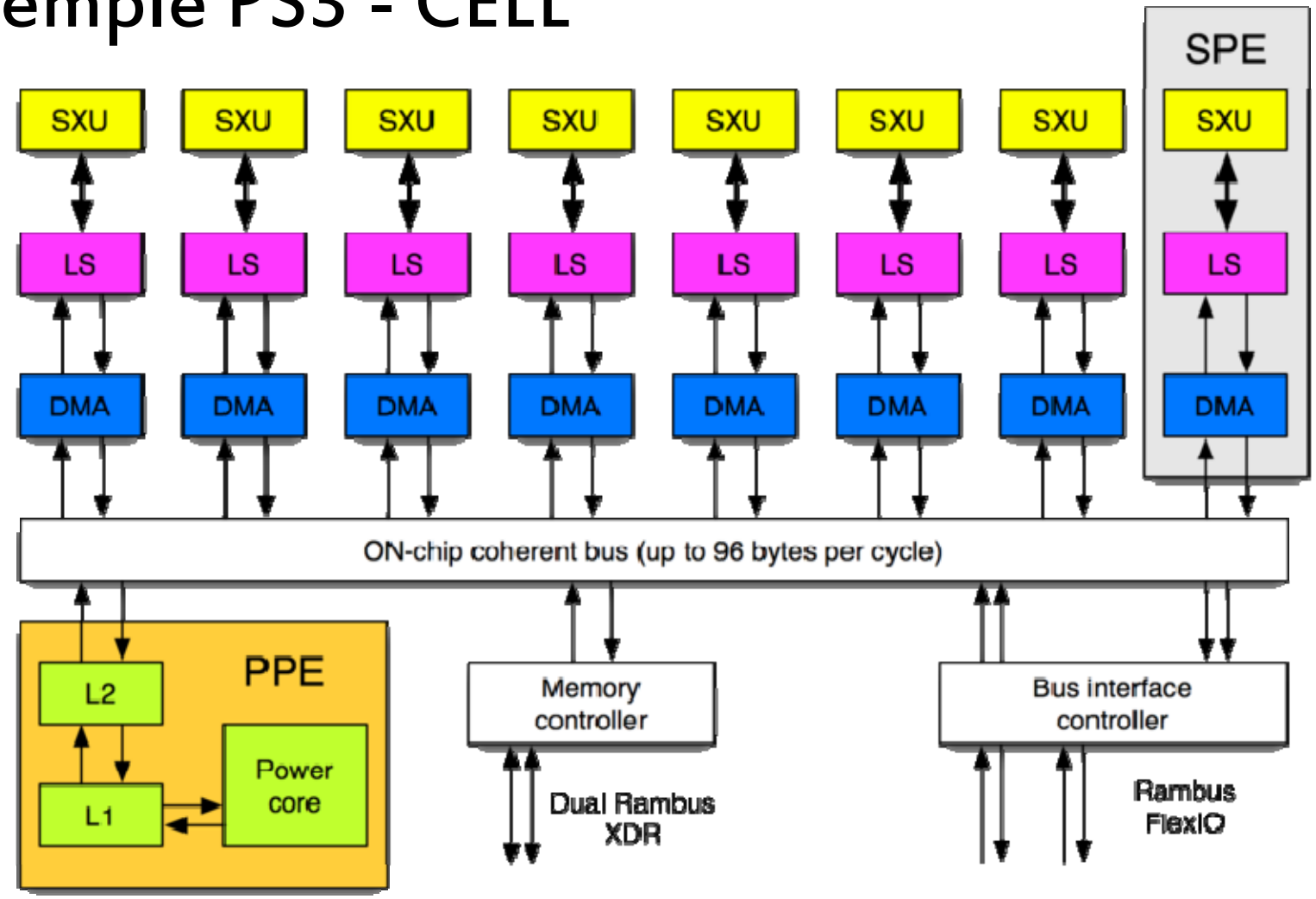
P: processeur demandeur, A: adresse demandée, D: contenu des données

Cohérence des mémoires – Mem. Distr.

- Machine à états
  - Bloc de cache individuel
  - Système à répertoire
  - Processeur local
    - Noir
  - Répertoire résident
    - Gris



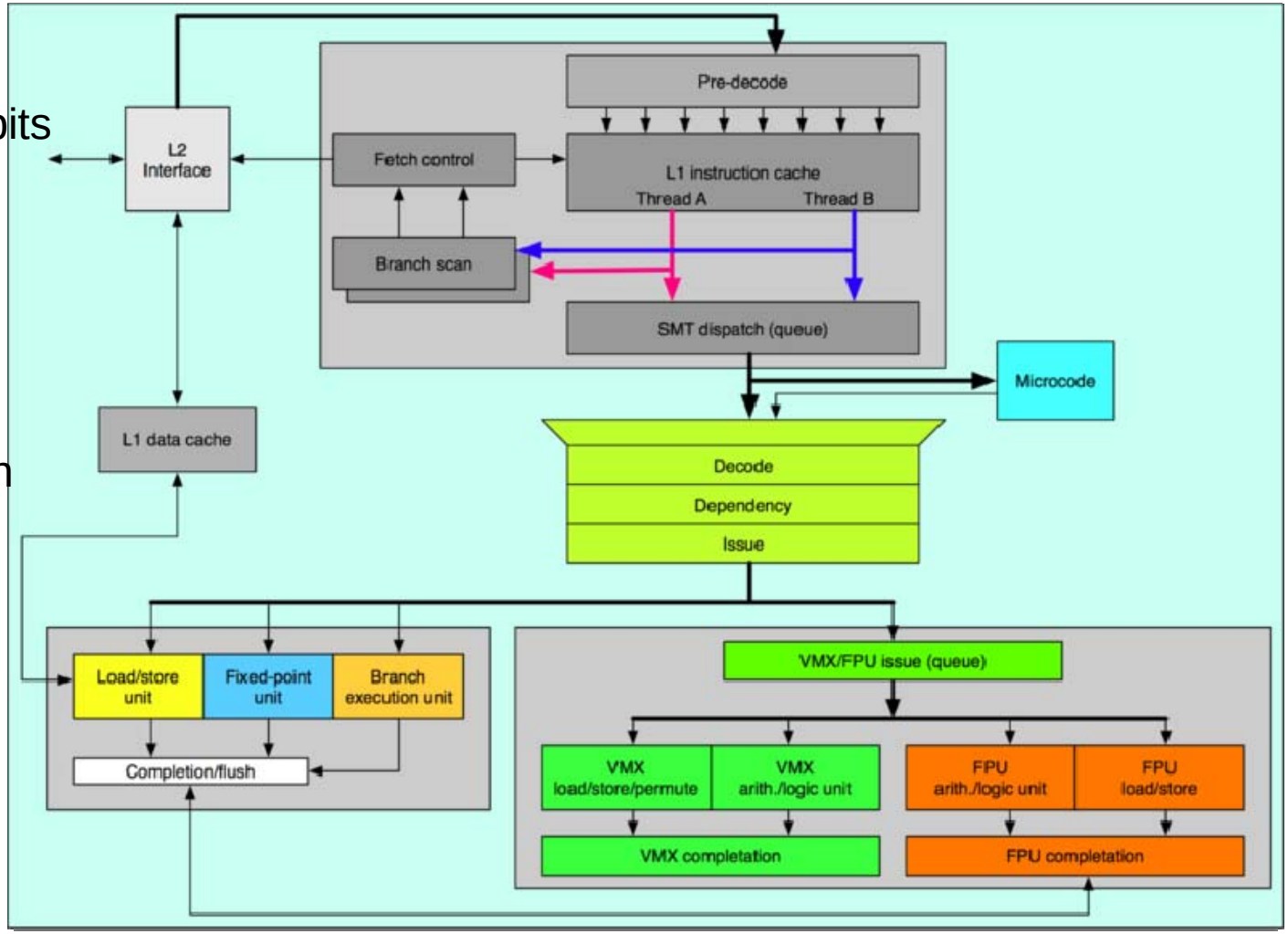
Exemple PS3 - CELL



PowerPC Processing Element ou PPE ,
Synergystic Processing Elements ou *SPE*,
Local Storage ou LS ,
Streaming Processor Unit ou SPU

Exemple PS3 – CELL - PPE

- Processeur 64 bits
- « in order »
- Distribue les instructions aux SPE
- Exécute l'instruction si un SPE ne peut le faire



Le CELL expliqué ici: *http://www.blachford.info/computer/Cell/Cell0_v2.html*

Exemple PS3 – CELL - SPE

- Processeur vectoriel
- SIMD RISC
- 4 unités calcul flottant DP
- 4 unités calcul entier