Architectures parallèles

Source: Michael J. Quinn
Parallel Programming in C with MPI and openMP

Plan

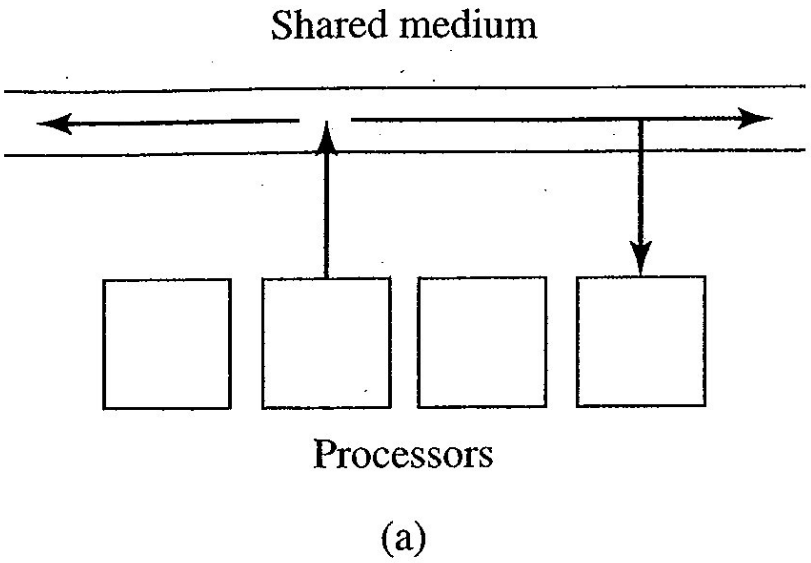
1. Réseaux d’intercommunication
2. Multiprocesseurs
3. Multi-ordinateurs
4. Ordinateurs vectoriels
5. Taxonomie de Flynn

1. Réseaux d’intercommunication

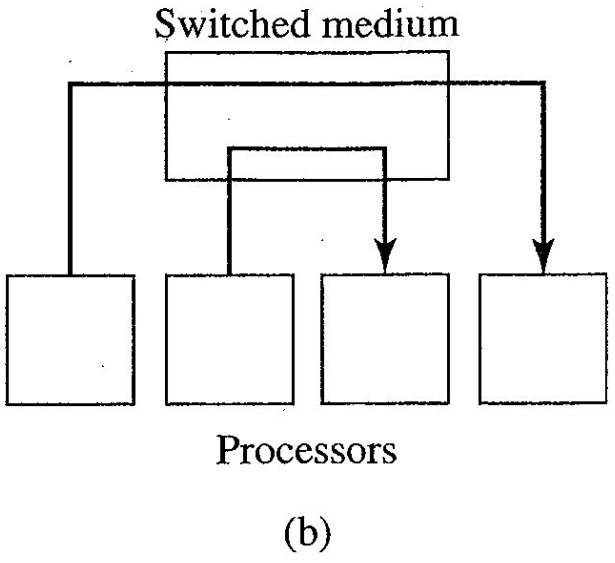
- Utilité
  - Connecter les processeurs à la mémoire partagée
  - Connecter les processeurs entre eux
- 2 modes:
  - Partagé
  - Avec commutateurs (switchs)

Comparaison des deux modes

Partagé



Commutateurs



Mode partagé

- Un seul message à la fois
- Les messages sont diffusés à tous les processeurs via un bus
- Chaque processeur “écoute” tous les messages
- L’arbitrage est nécessaire
- Une collision nécessite la rediffusion du message
- Limite le nombre de processeurs

Avec commutateurs

- Permet la communication point-à-point
- Avantages:
  - Permet l'envoi de plusieurs messages simultanément
  - Permet l'utilisation d'un plus grand nombre de processeurs
- Conçu avec des circuits *crossbar*

Circuit crossbar

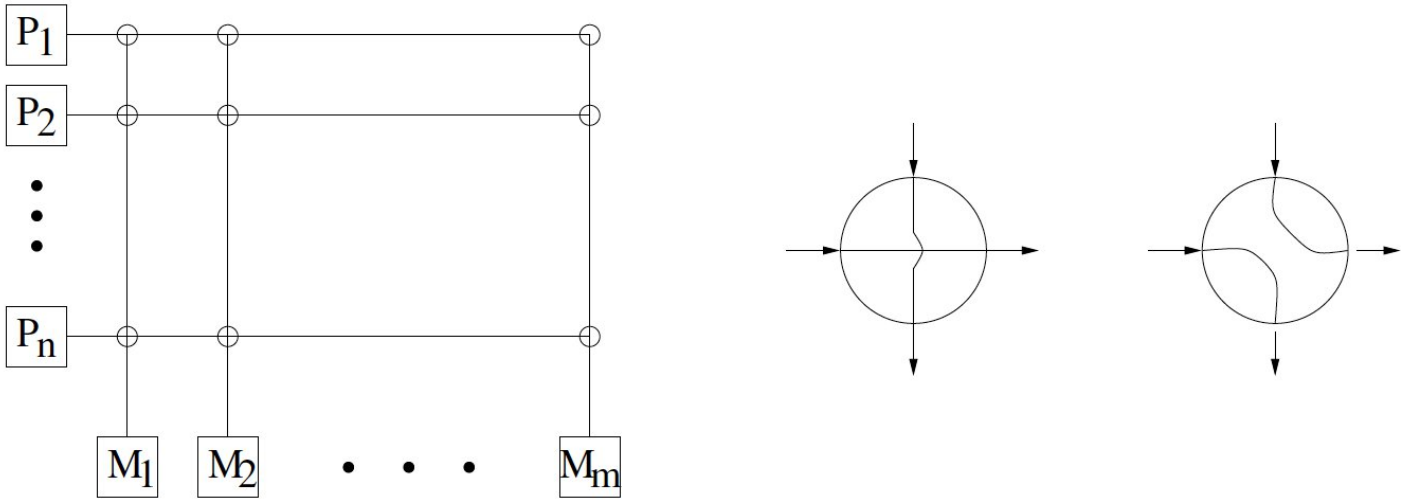


Fig. 2.14 Illustration of a $n \times m$ crossbar network for n processors and m memory modules (left). Each network switch can be in one of two states: straight or direction change (right).

Image tirée de: Thomas Rauber et Gudula Rünger,
Parallel programming for multicore and cluster systems, 2023.

Topologies des réseaux d'intercommunication

- Représentation sous forme de graphe
  - Noeud = processeur ou commutateur
  - Arête = lien de communication
- Deux types de topologies
  - Directe
  - Indirecte

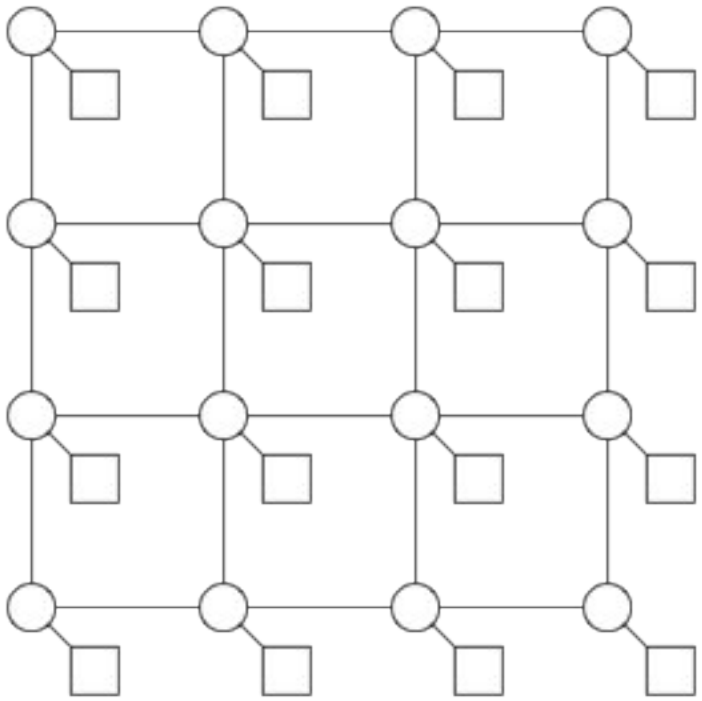
Topologie directe

- Un commutateur pour chaque processeur
- Chaque commutateur est connecté à:
  - 1 processeur
  - Au moins un autre commutateur
- En général, le nombre de processeurs est identique au nombre de commutateurs
- Les connections entre processeurs est fixe

Exemple: Grille 2-D

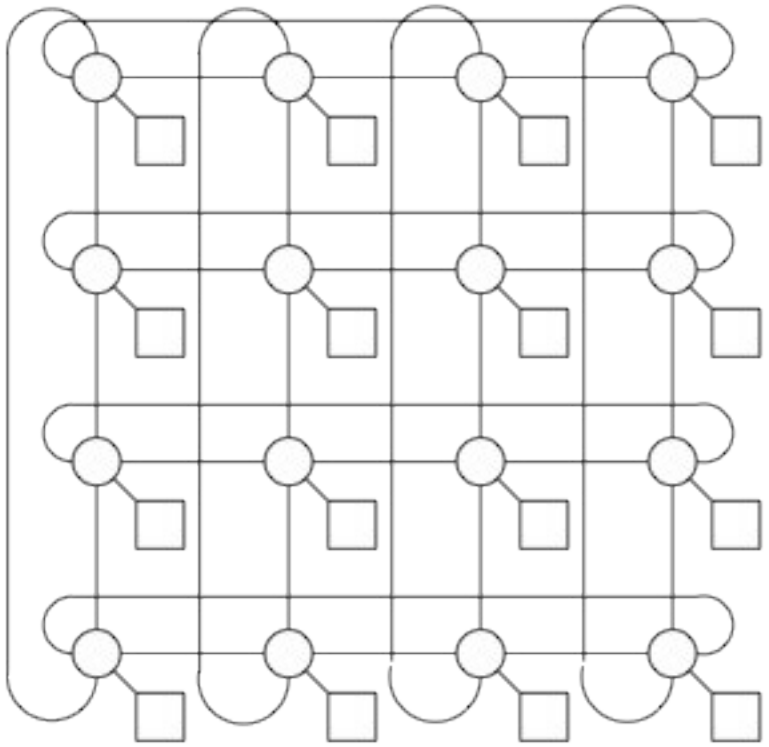
- Topologie directe
- Les commutateurs sont organisés sous la forme d’une grille 2-D
- Communication permise seulement entre les noeuds voisins
- Tore: les extrémités sont reliées

Grille 2-D



(a)

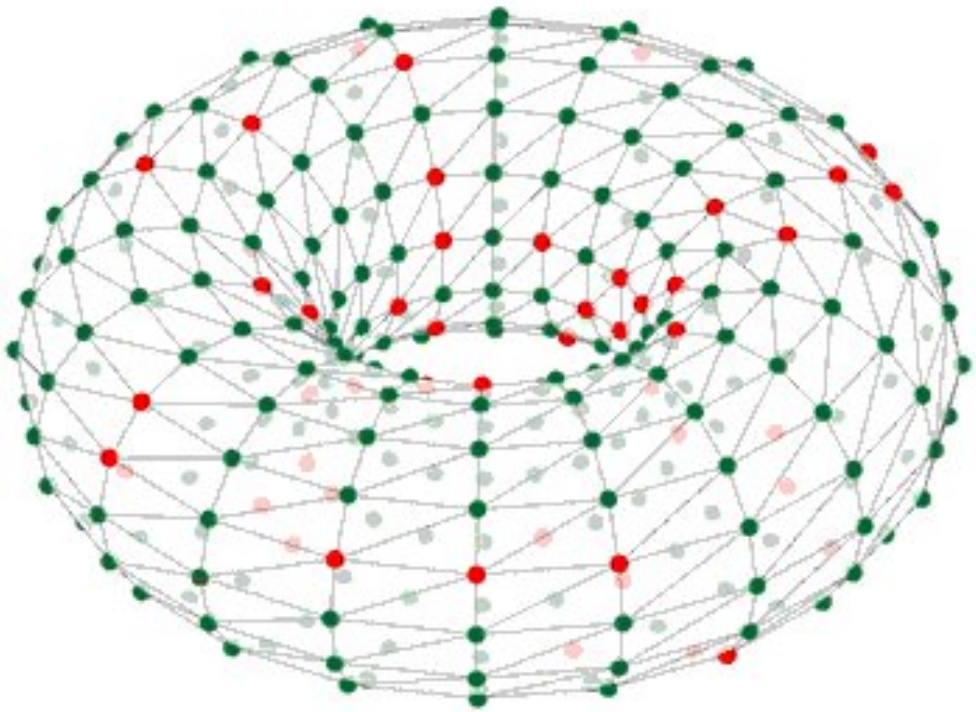
Tore



(b)

Les cercles représentent des commutateurs et les carré des processe

Tore



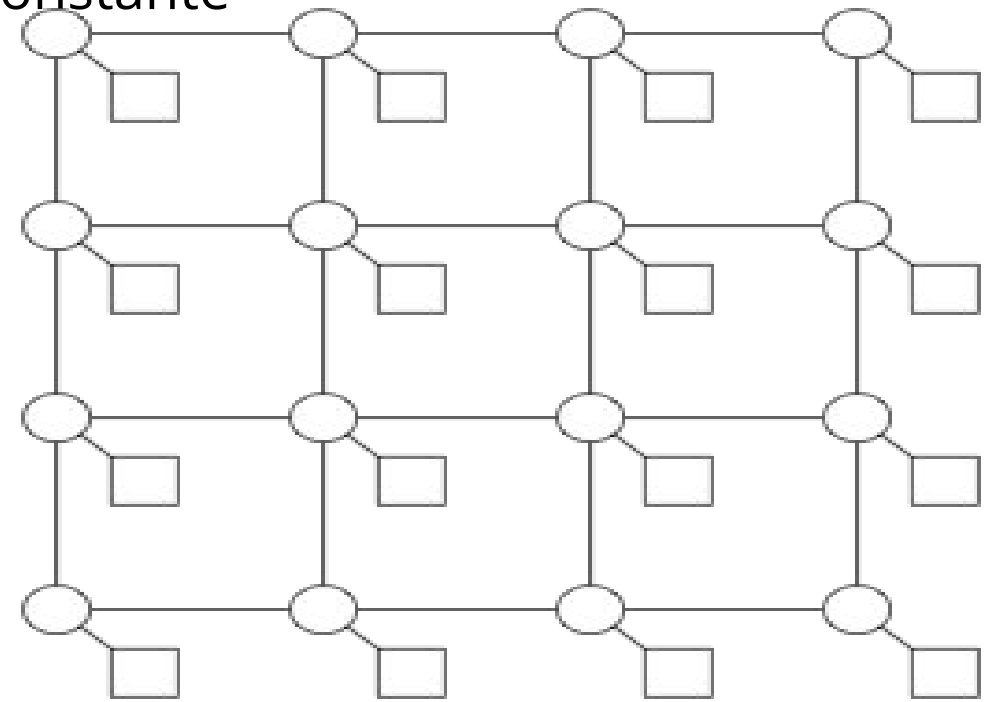
Évaluation d'une topologie

- Diamètre
  - Distance maximale entre deux noeuds
  - Borne inférieure sur le temps de communication
- Largeur de coupe
  - Nombre min d'arêtes à enlever pour diviser le graphe en deux composantes connexes de même taille (à 1 neud près).
  - Borne supérieure sur le nombre de messages pouvant être envoyés simultanément.
- Degré = Nombre d'arêtes adjacentes à un noeud
- Longueur des arêtes:

  Une longueur constante permet plus facilement d'augmenter le nombre de processeurs.

Évaluation des grilles 2-D

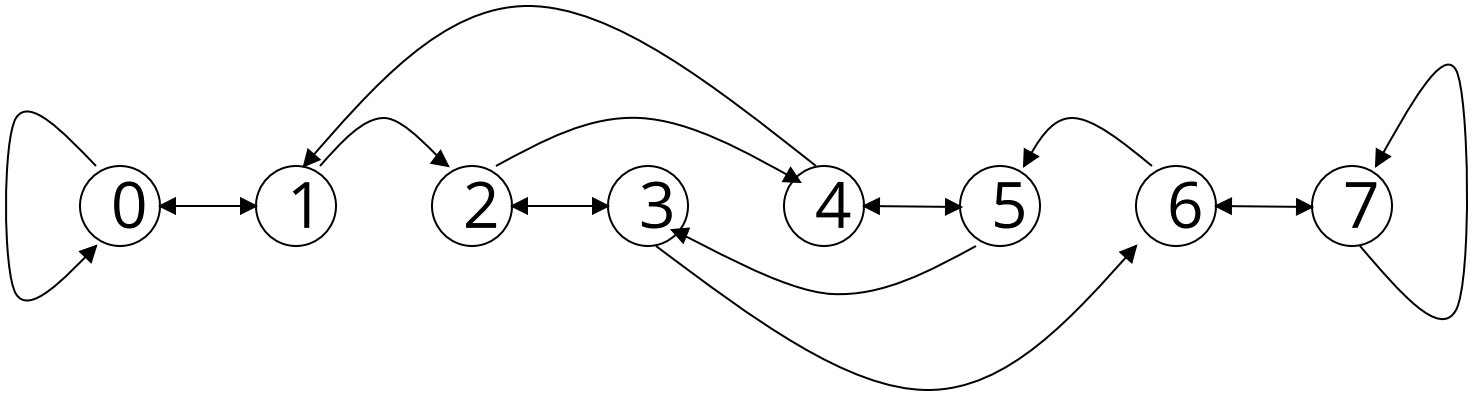
- Diamètre: $(n^{1/2})$
- Largeur de coupe: $(n^{1/2})$
- Degré: 4
- Longueur d'arêtes constante



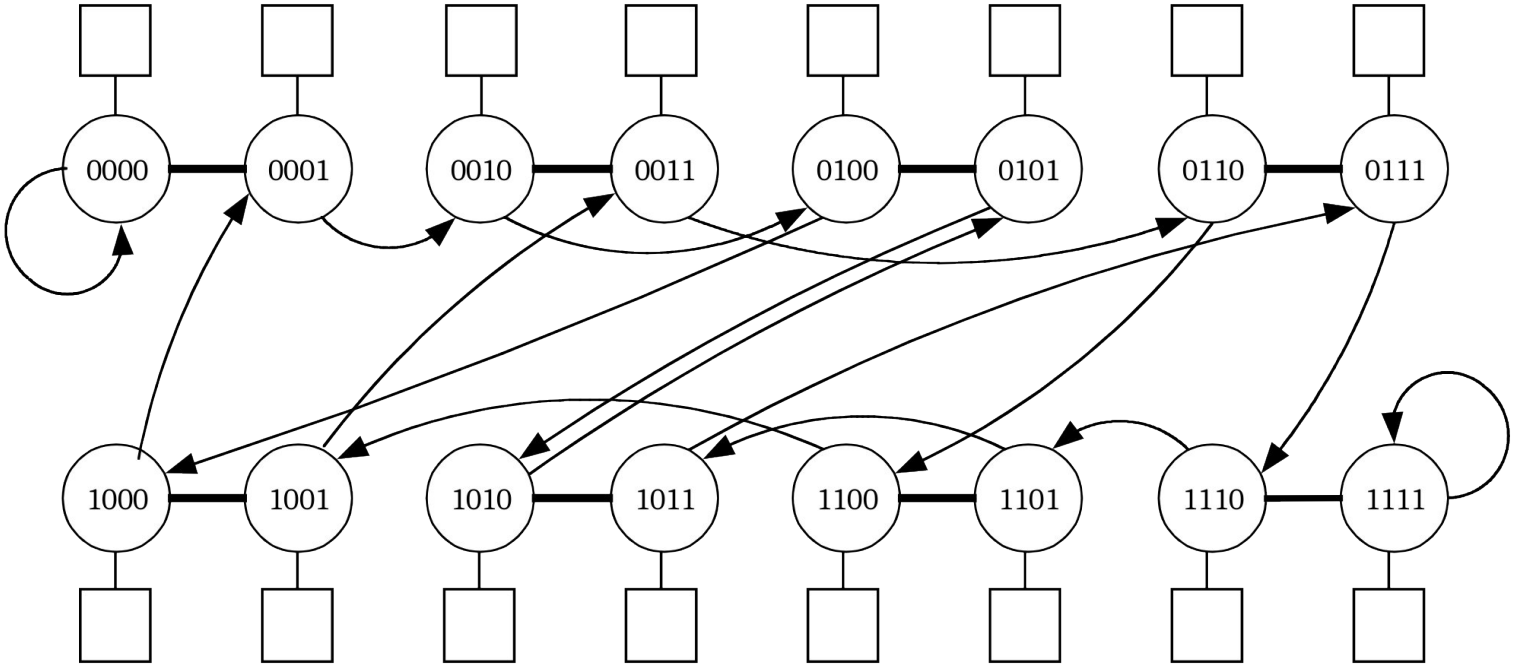
Réseau shuffle-exchange

- Topologie directe
- Le nombre de noeuds est une puissance de 2
- Adresses: 0, 1, …, 2^k-1
- Deux arcs sortent de chaque noeud *i*
  - (i,k): où k est la rotation à gauche des bits de i
  - (i,j): où i et j ne diffèrent qu'au bit le moins significatif

Illustration du shuffle-exchange



Adressage du shuffle-exchange



Exemple

0101 à 1110
0100 : 1110 (exchange)
1000 : 1101 (shuffle)
1001 : 1101 (exchange)
0011 : 1011 (shuffle)
0011 : 1011 (pas d’exchange)
0110 : 0111 (shuffle)
0111 : 0111 (exchange)
1110 : 1110 (shuffle)

Évaluation du shuffle-exchange

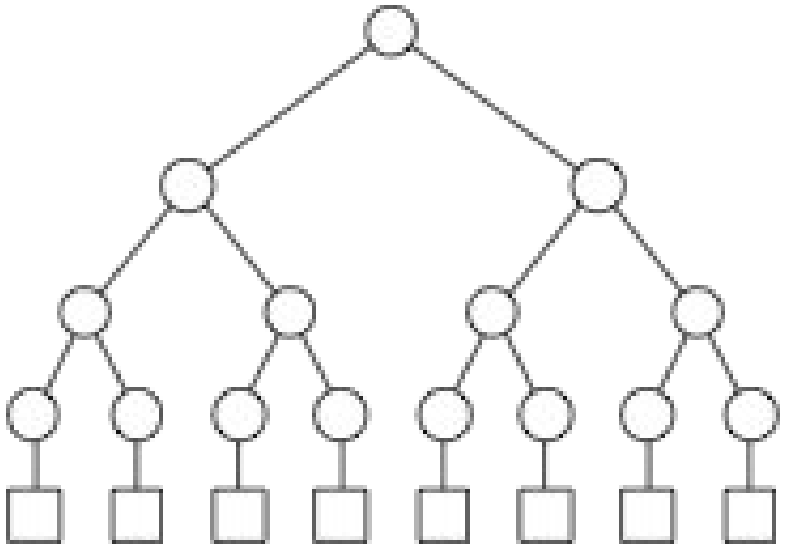
- Diamètre: $2\log n - 1$
- Largeur de coupe: $n / \log n$
- Degré: 2
- Longueur d'arêtes non constante

Topologie indirecte

- Le nombre de commutateurs peut être plus grand que le nombre de processeurs
- Certains commutateurs ne sont connectés qu'à d'autres commutateurs
- Les connexions entre les processeurs est dynamique.

Arbre binaire

- Topologie indirecte
- $n = 2^d$ processeurs
- $n-1$ commutateurs



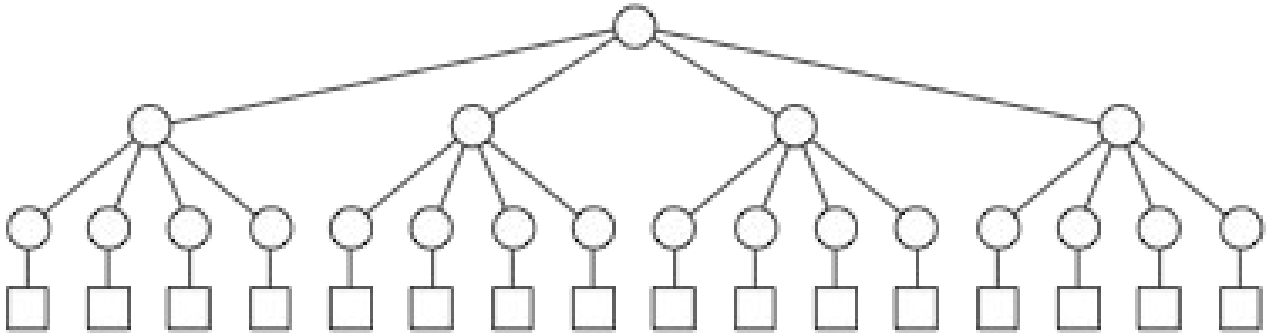
Évaluation d’un arbre binaire

- Diamètre: 2 log n
- Largeur de coupe: 1
- Degré: 3
- La longueur d’arête n’est pas constante

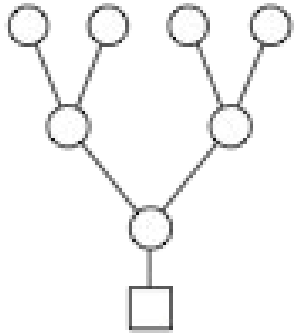
Hyper-arbre

- Topologie indirecte
- Faible diamètre
- Plus grande largeur de coupe qu'un arbre
- De face, il apparaît comme un arbre de degré k et profondeur d
- De côté, il apparaît comme un arbre binaire renversé de hauteur d

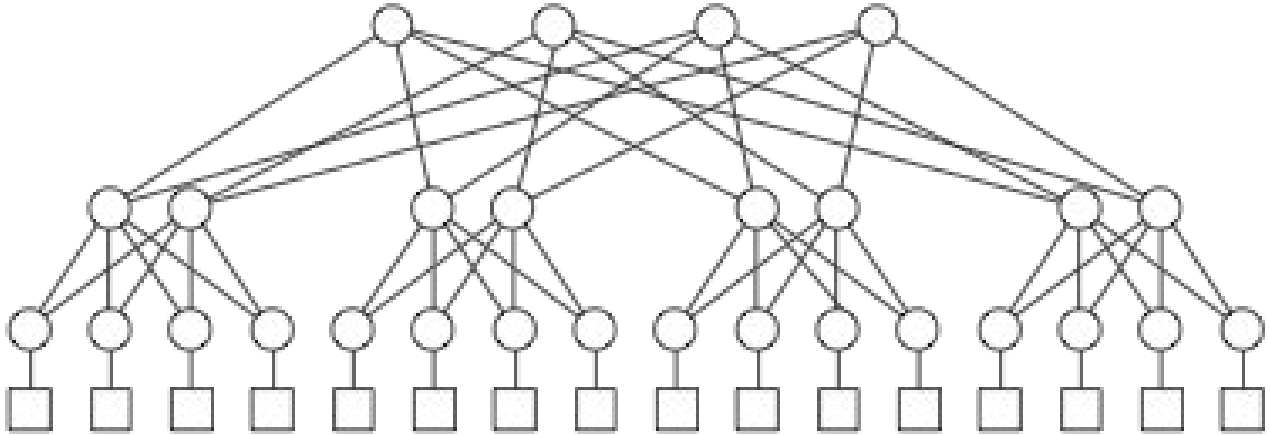
Hyper-arbre



(a)



(b)



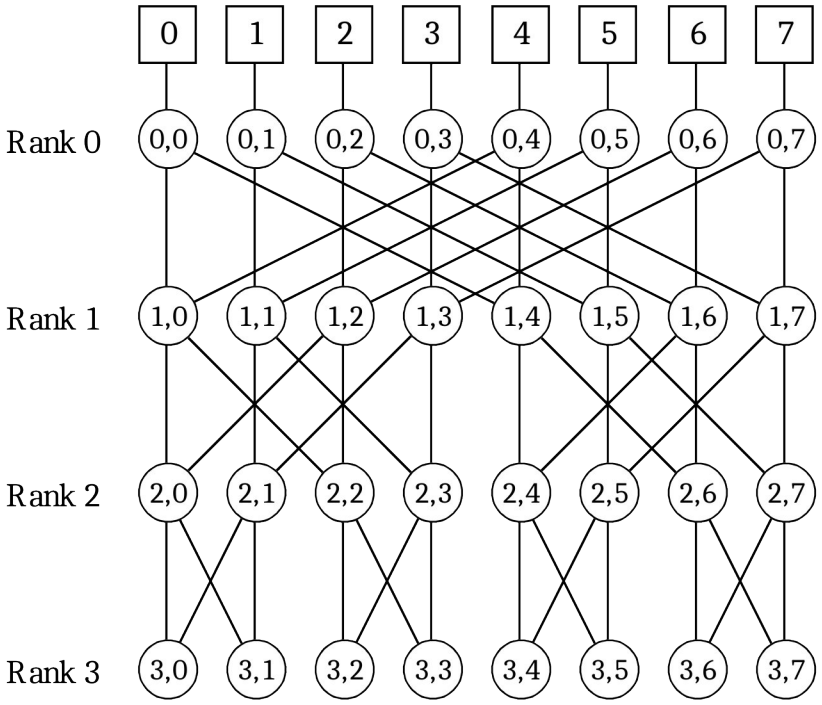
(c)

Évaluation des hyper-arbre

- Diamètre: $2d = 2\log_k n$
- Largeur de coupe: $(k/2)2^d$
- Degré: k+2
- Longueur d'arêtes non constante

Réseau butterfly

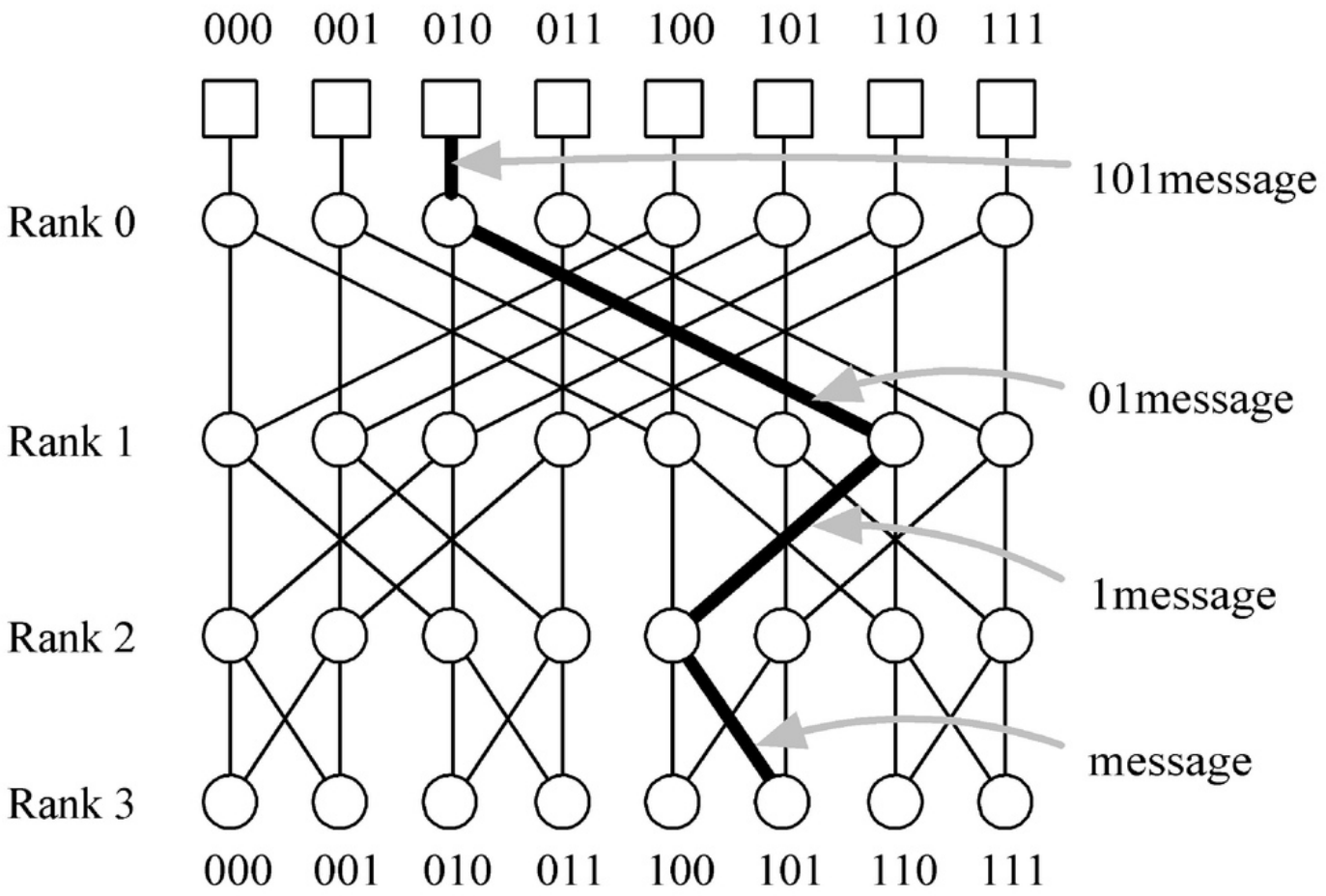
- Topologie indirecte
- $n = 2^d$ processeurs connectés à l'aide de $n(\log n + 1)$ commutateurs.



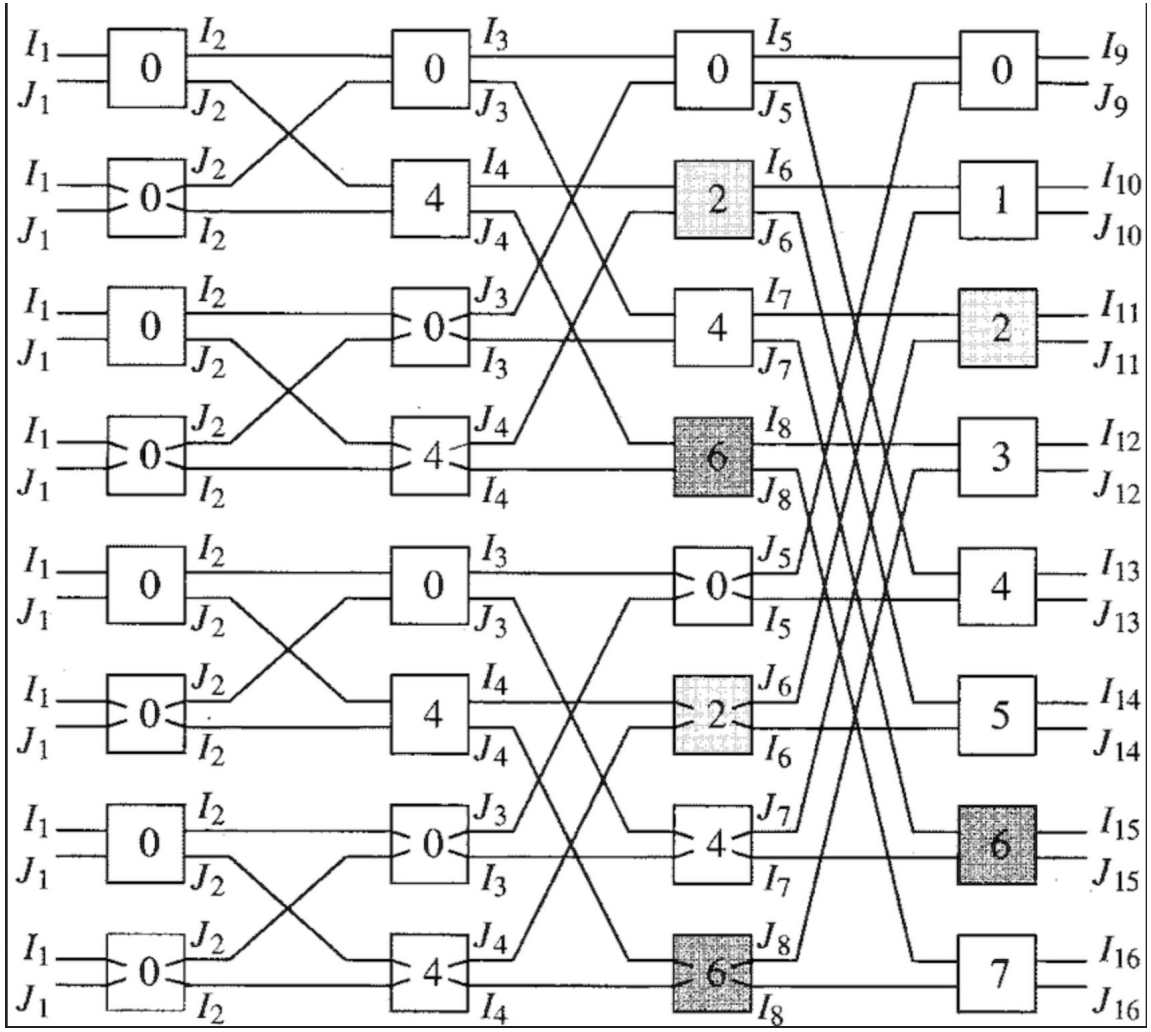
Évaluation

- Diamètre: log n
- Largeur de coupe: n / 2
- Degré: 4
- Longueur d’arêtes non constante

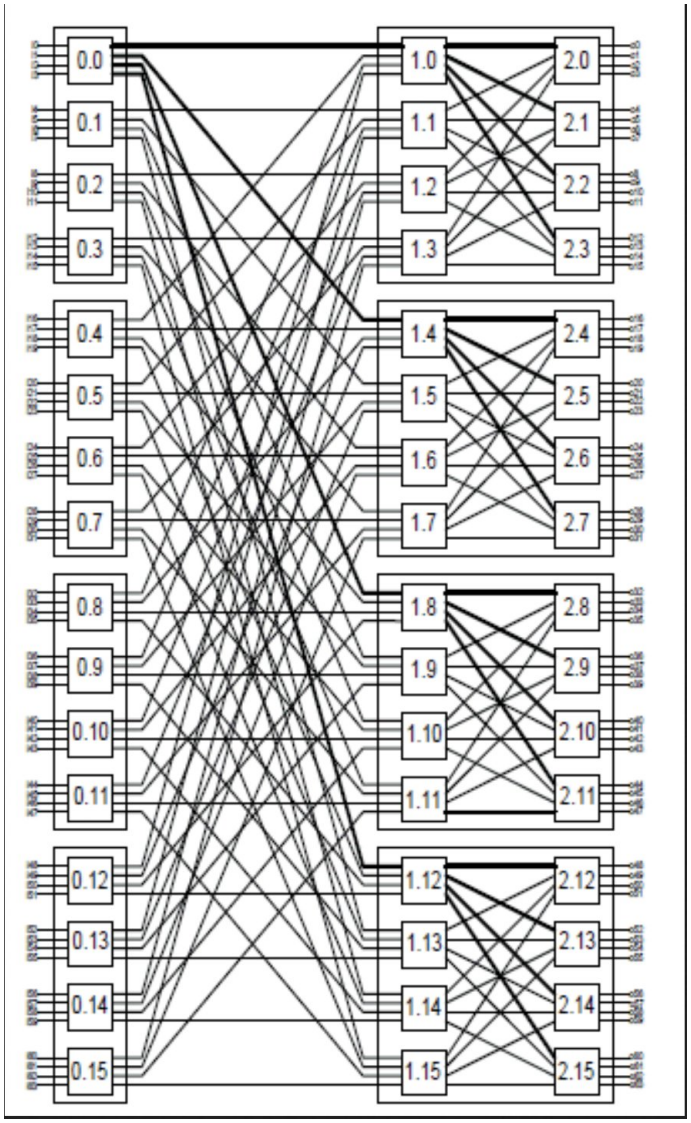
Routage sur un réseau butterfly



Réseau butterfly



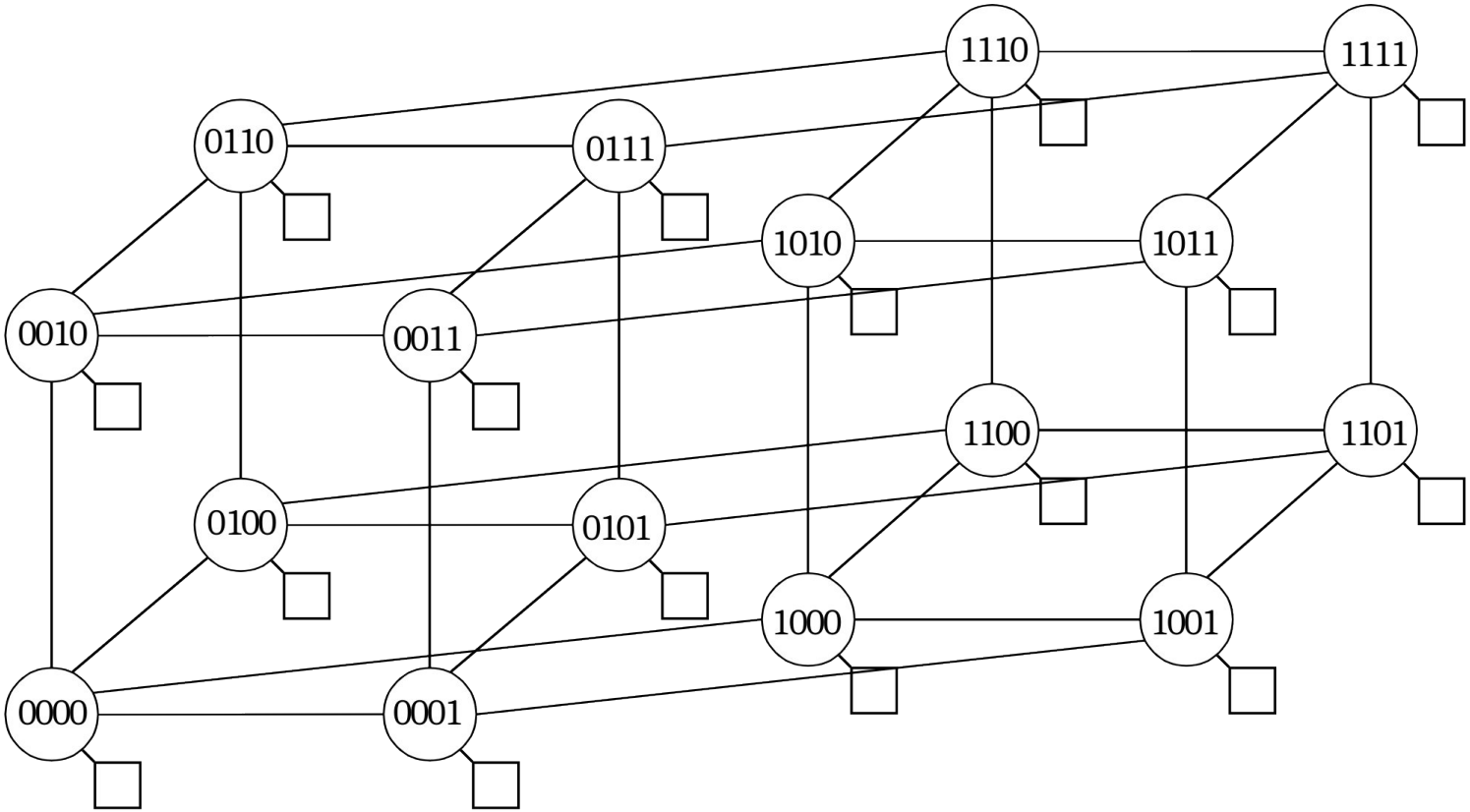
Réseau butterfly



Hypercube

- Topologie directe
- Le nombre de noeuds est un puissance de 2
- Adresses: 0, 1, …, 2^k-1
- Le noeud i est connecté au noeud k si et seulement si leurs adresses ne diffèrent que d'un seul bit.

Réseau hypercube de 16 processeurs



Évaluation

- Diamètre: log n
- Largeur de coupe: n / 2
- Degré: log n
- La longueur des arêtes n’est pas constante

Comparaison des réseaux

- Tous ceux que nous avons vus ont un diamètre logarithmique sauf la grille 2-D
- Hyper-arbre, butterfly et hypercube ont une largeur de coupe $n / 2$
- Tous ont un degré constant sauf l'hypercube
- Seul la grille 2-D a une longueur d'arête constante.

2. Multiprocesseurs

- Plusieurs processeurs utilisant une mémoire commune
- Même espace d’adressage
- Peuvent être construits avec des processeurs standards

Multiprocesseurs

- Deux types de multiprocesseurs:
  - Mémoire centralisées
  - Mémoire distribuées

Multiprocesseurs centralisés

- Extension directe des monoprocesseurs
- Plusieurs processeurs connectés à un même bus
- Tous les processeurs partagent une mémoire commune
- Le temps d'accès à la mémoire est identique pour tous les processeurs
  - *Uniform memory access* (UMA)
  - *Symmetrical multiprocessor (SMP)*
- Chaque processeur possède sa propre mémoire cache

Multiprocesseur avec bus et mémoire centralisée (UMA)

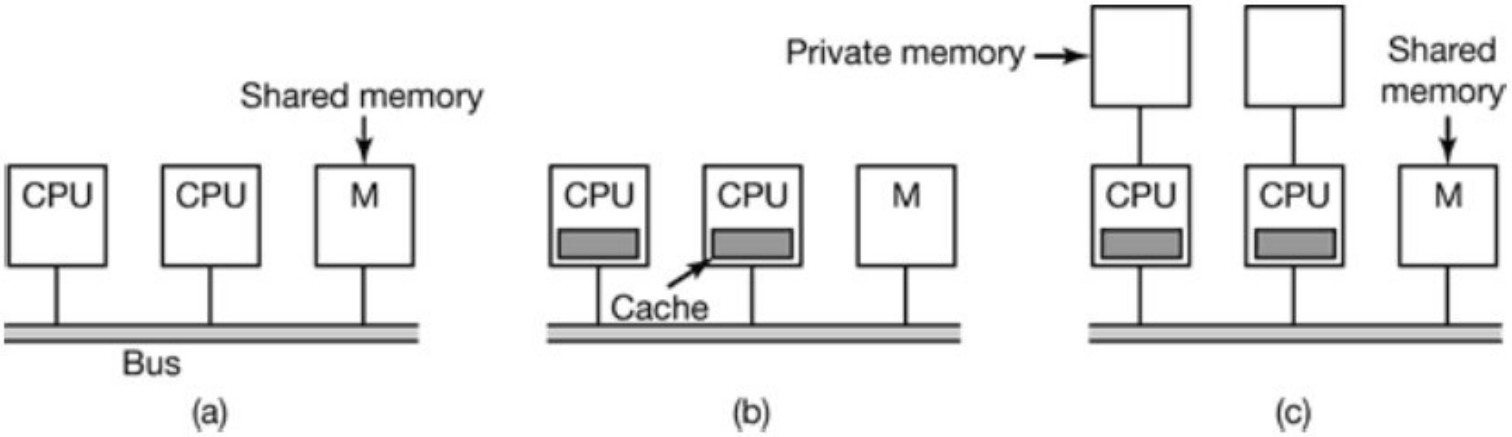


Figure 8-2. Three bus-based multiprocessors. (a) Without caching. (b) With caching. (c) With caching and private memories.

(a) Sans cache
(b) Avec caches
(c) Avec caches et mémoire privée

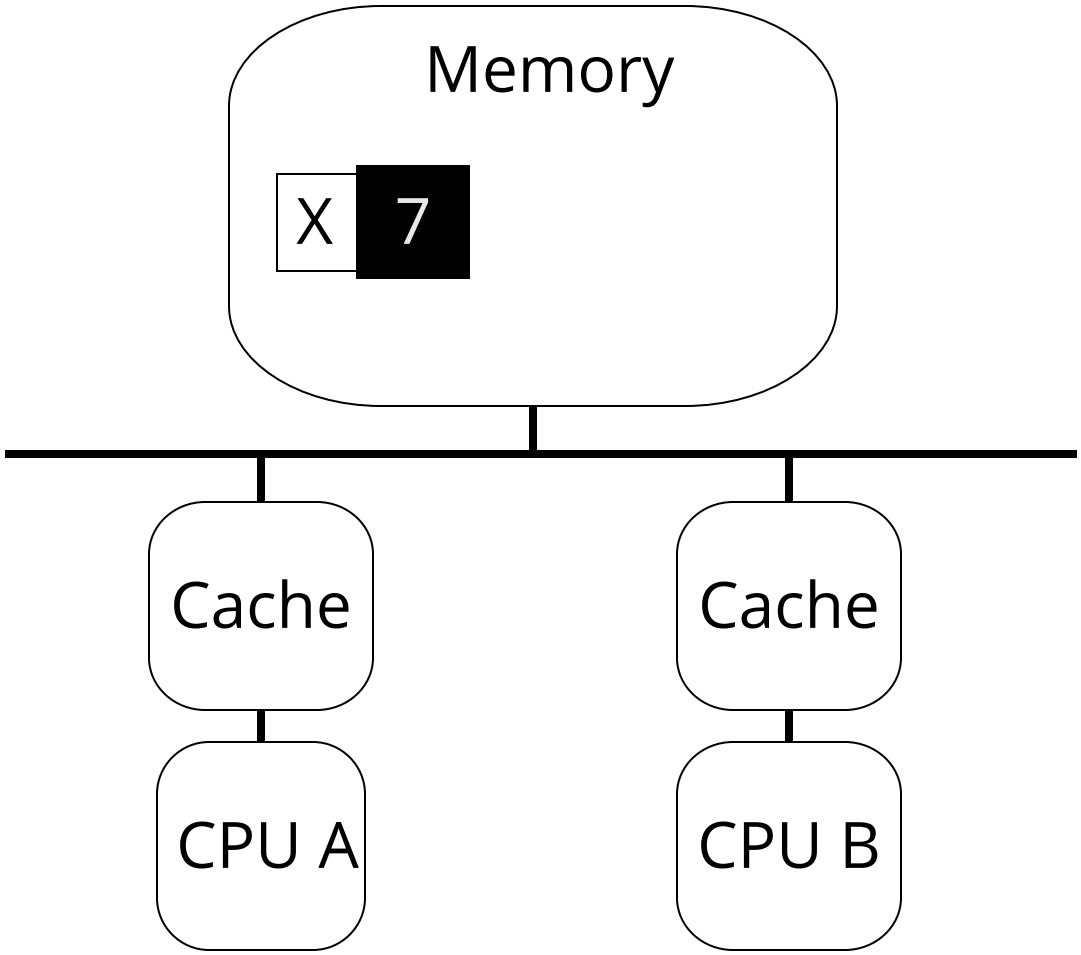
Données privées et partagées

- Données privées: Accessible à un seul processeur
- Données partagées: Accessible à tous les processeurs
- La communication entre les processeurs se fait à l'aide des données partagées.

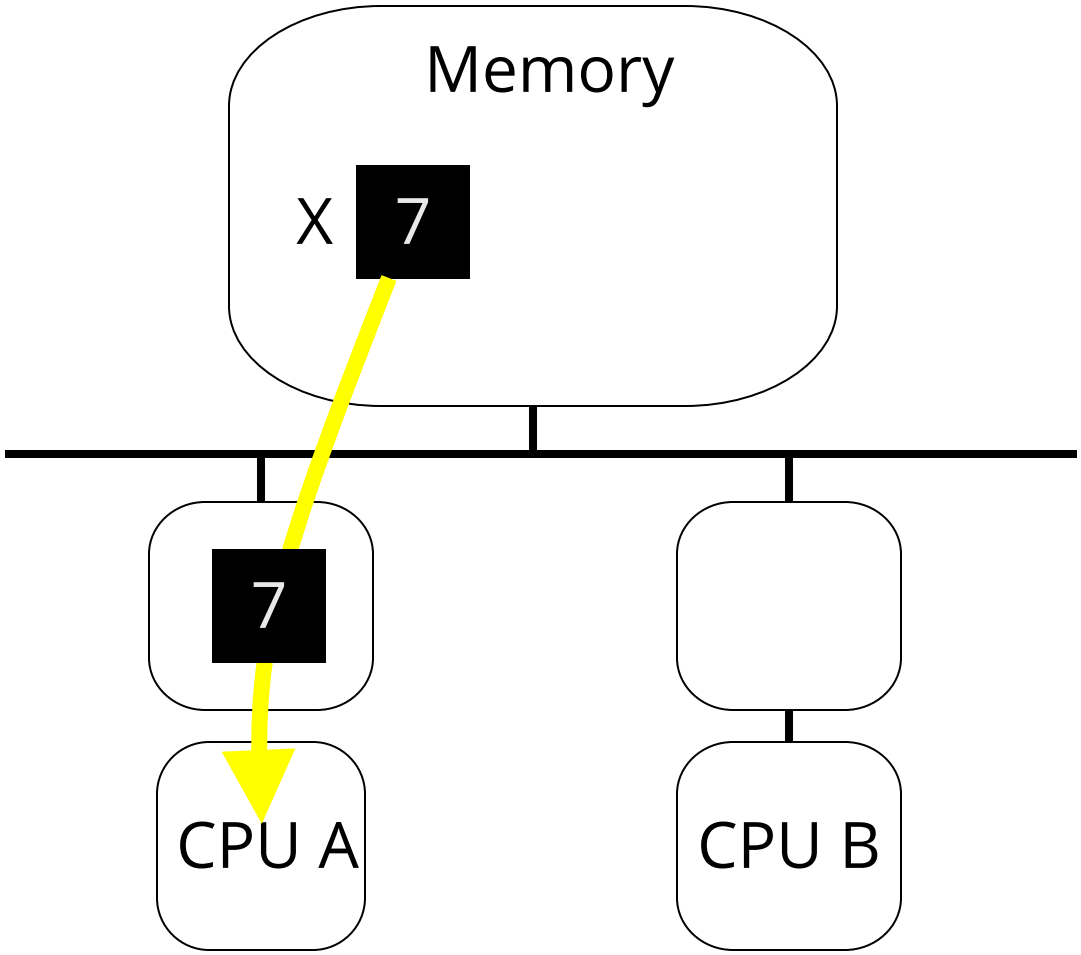
Problèmes liés aux variables partagées

- Cohérence des caches
  - La présence de caches réduit les problèmes de bande passante
  - Comment s'assurer que différents processeurs possèdent la même valeur pour une variable partagée?
- Synchronisation
  - Exclusion mutuelle
  - Barrière

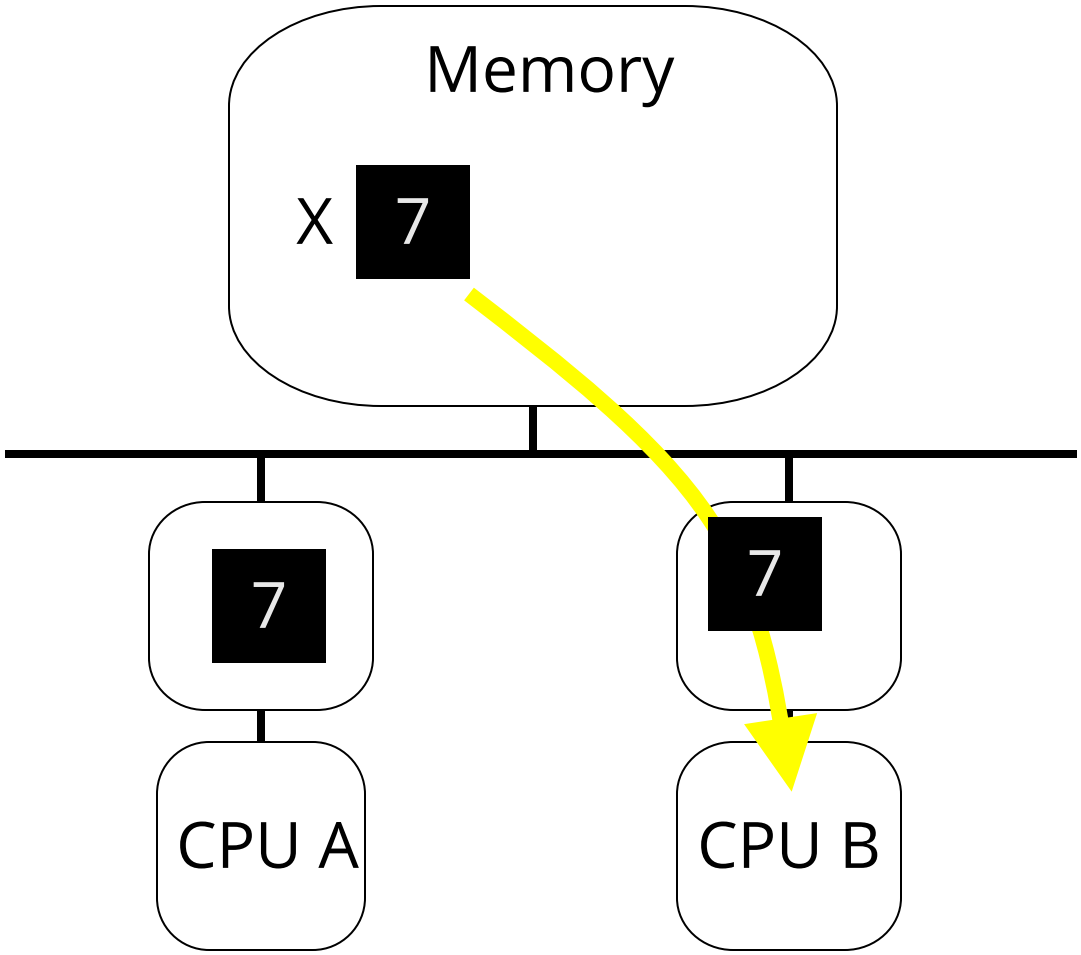
Cohérence des caches



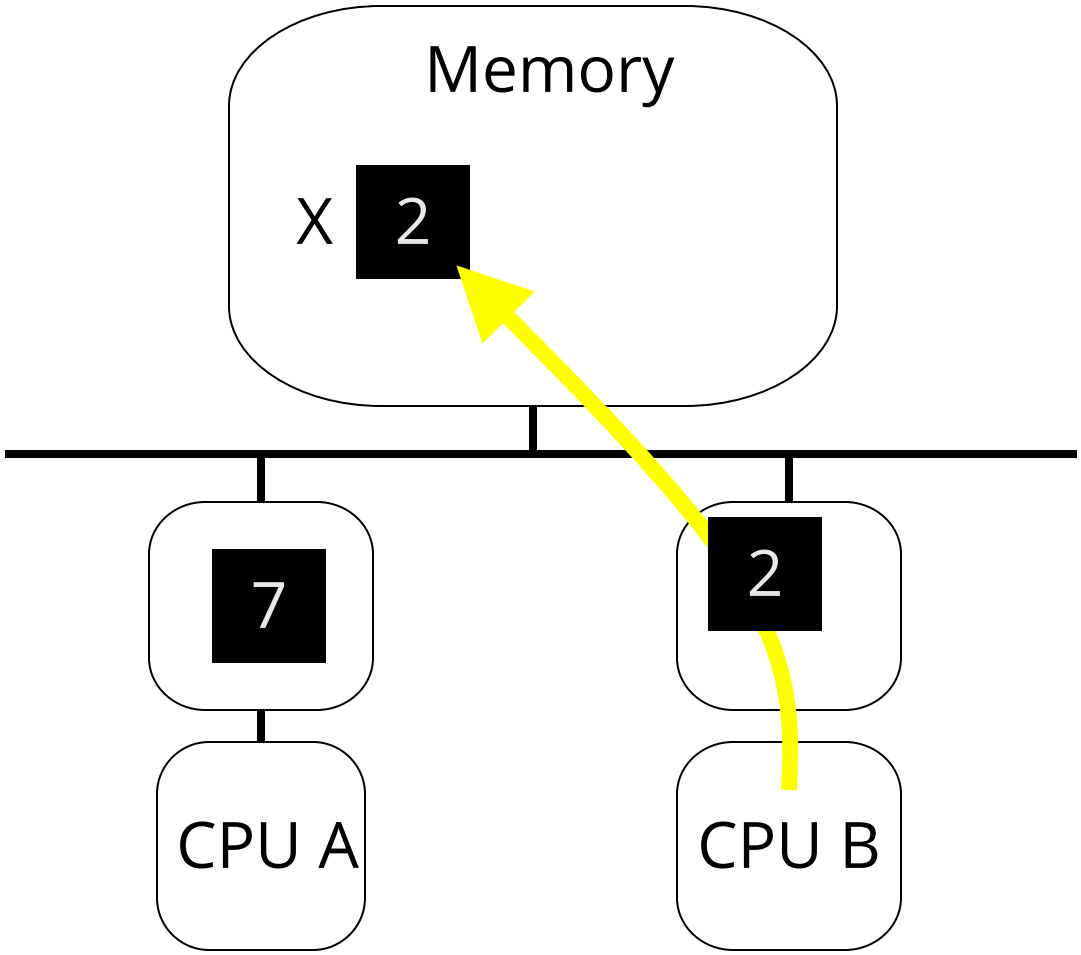
Cohérence des caches



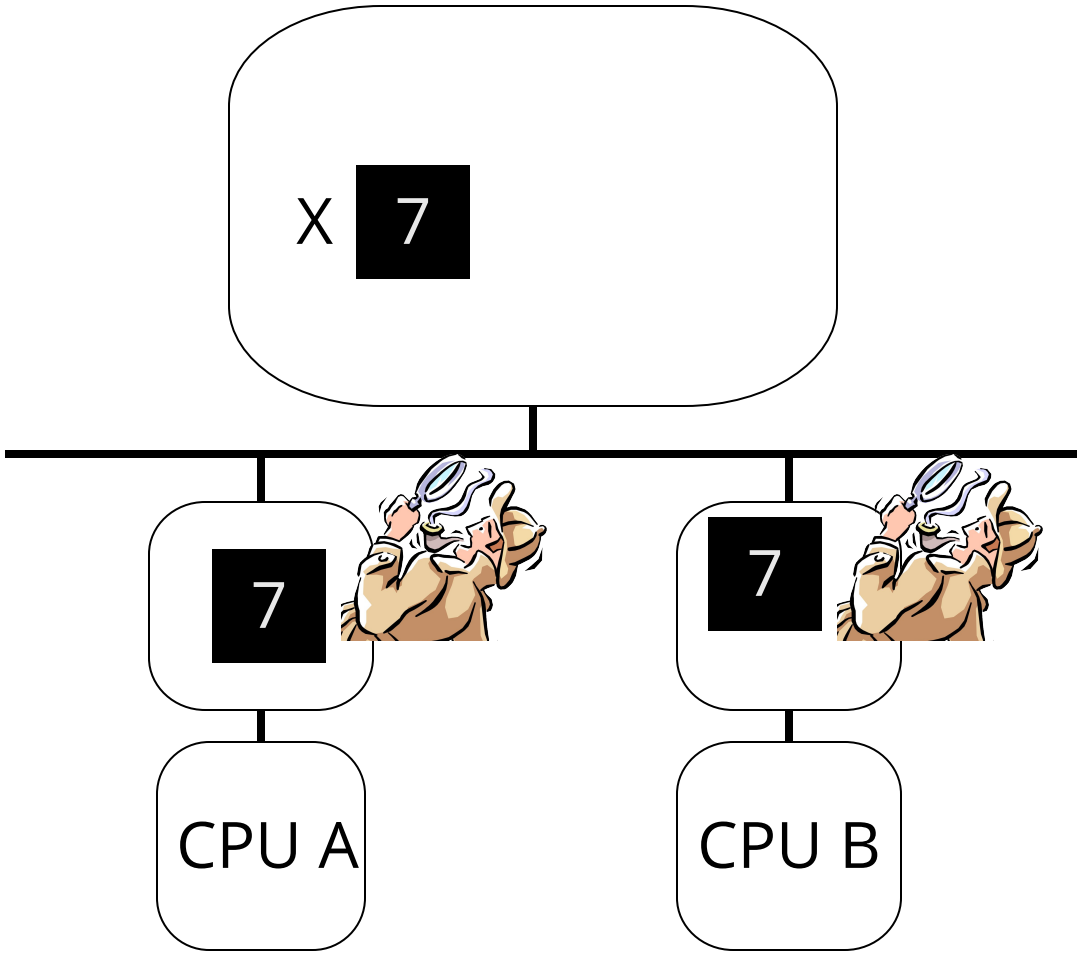
Cohérence des caches



Cohérence des caches

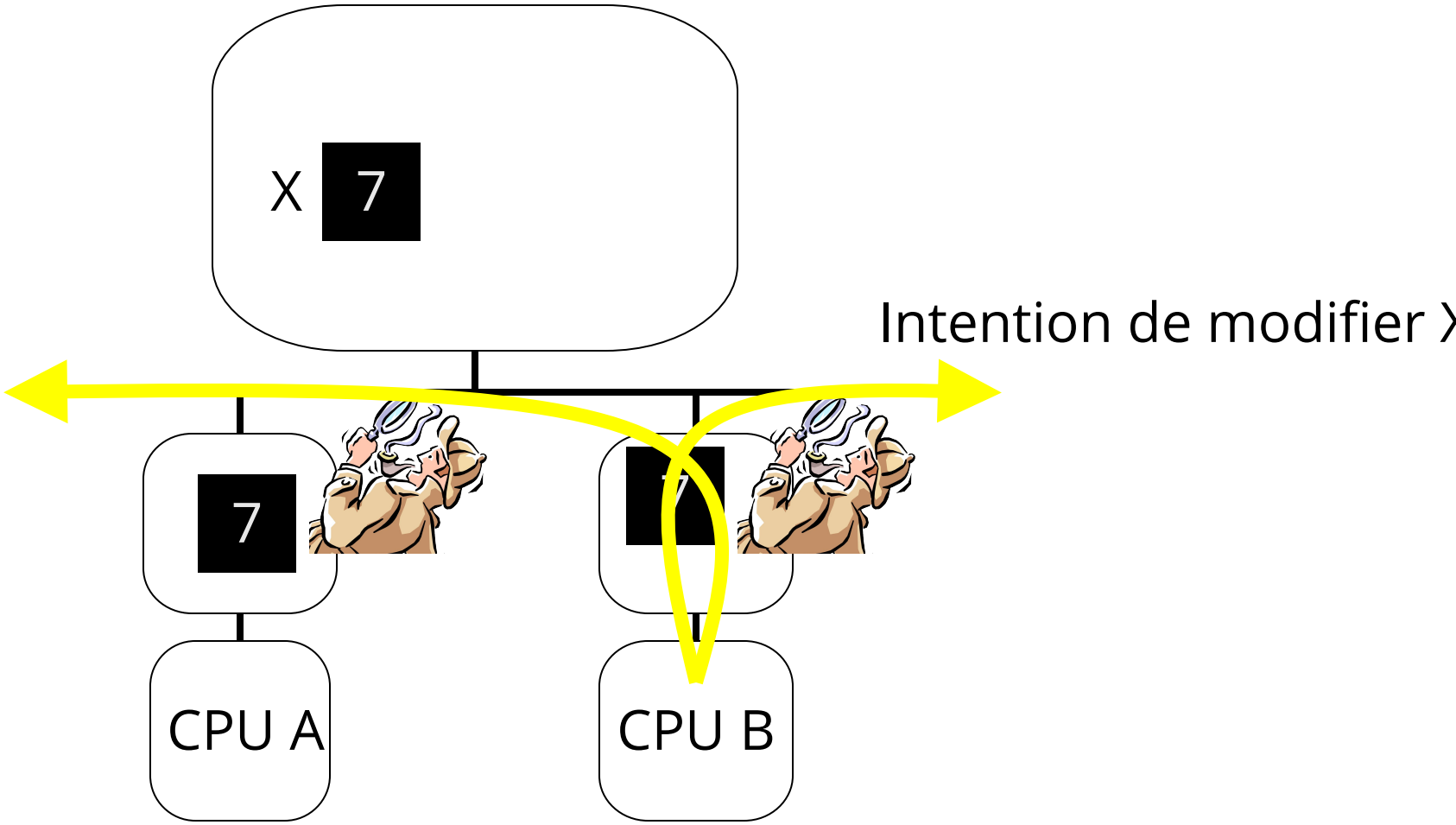


Protocole d’invalidation en écriture

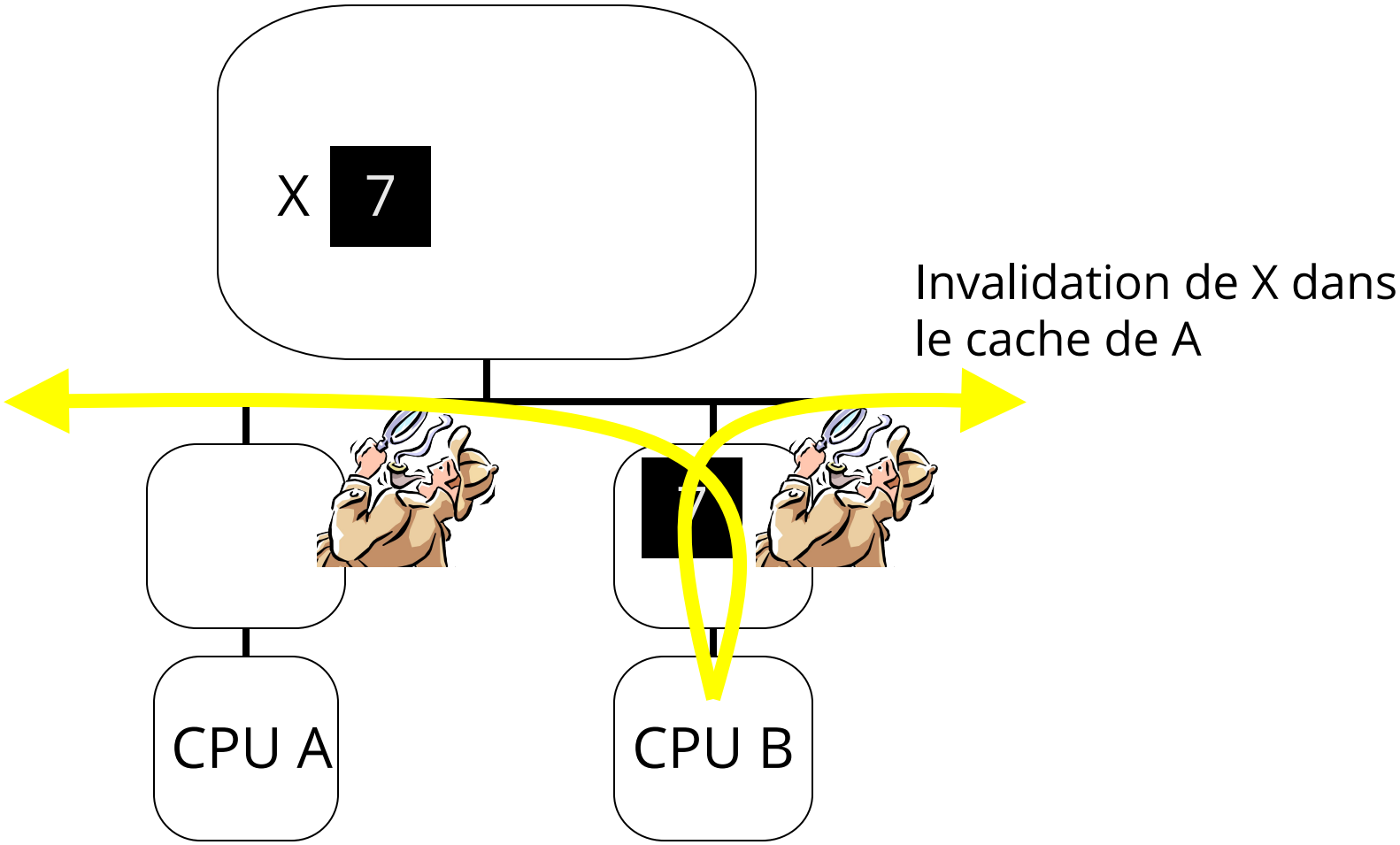


Chaque cache possède un moniteur de bus

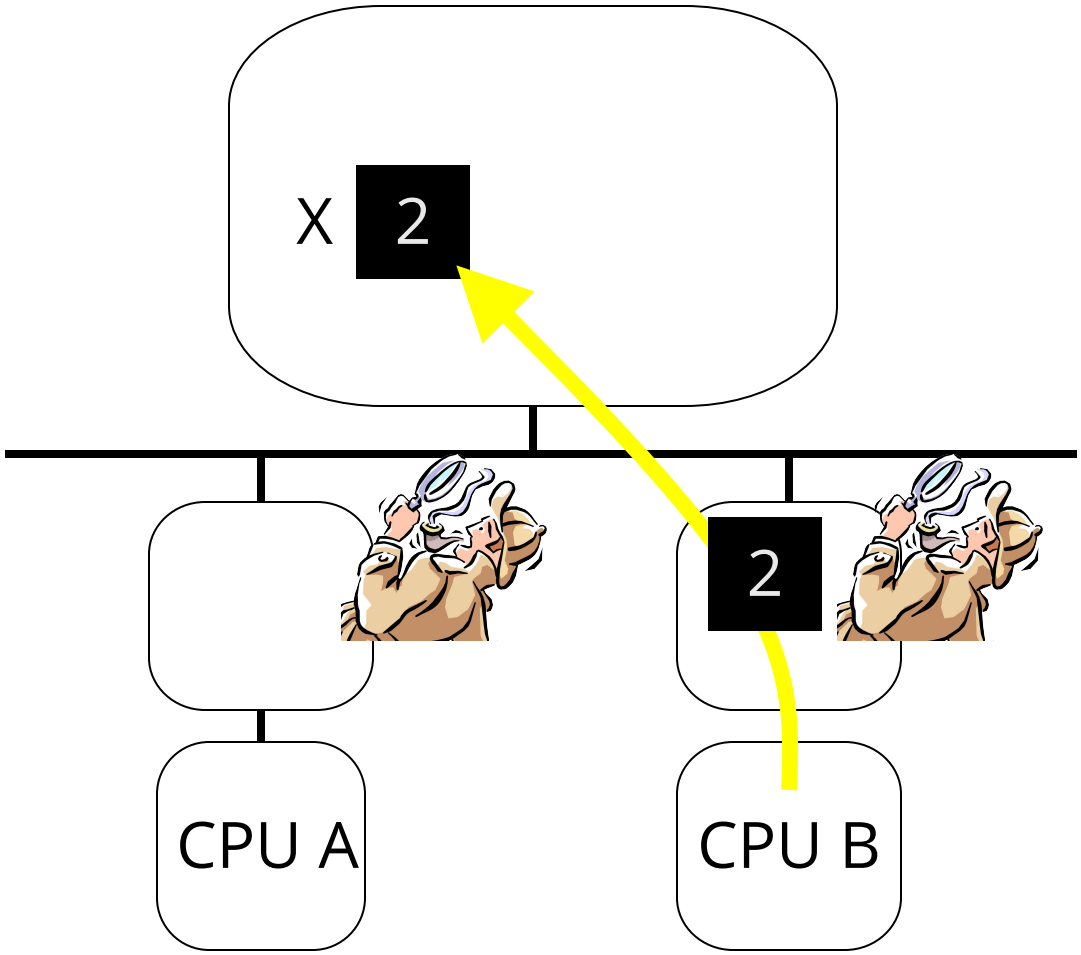
Protocole d’invalidation en écriture



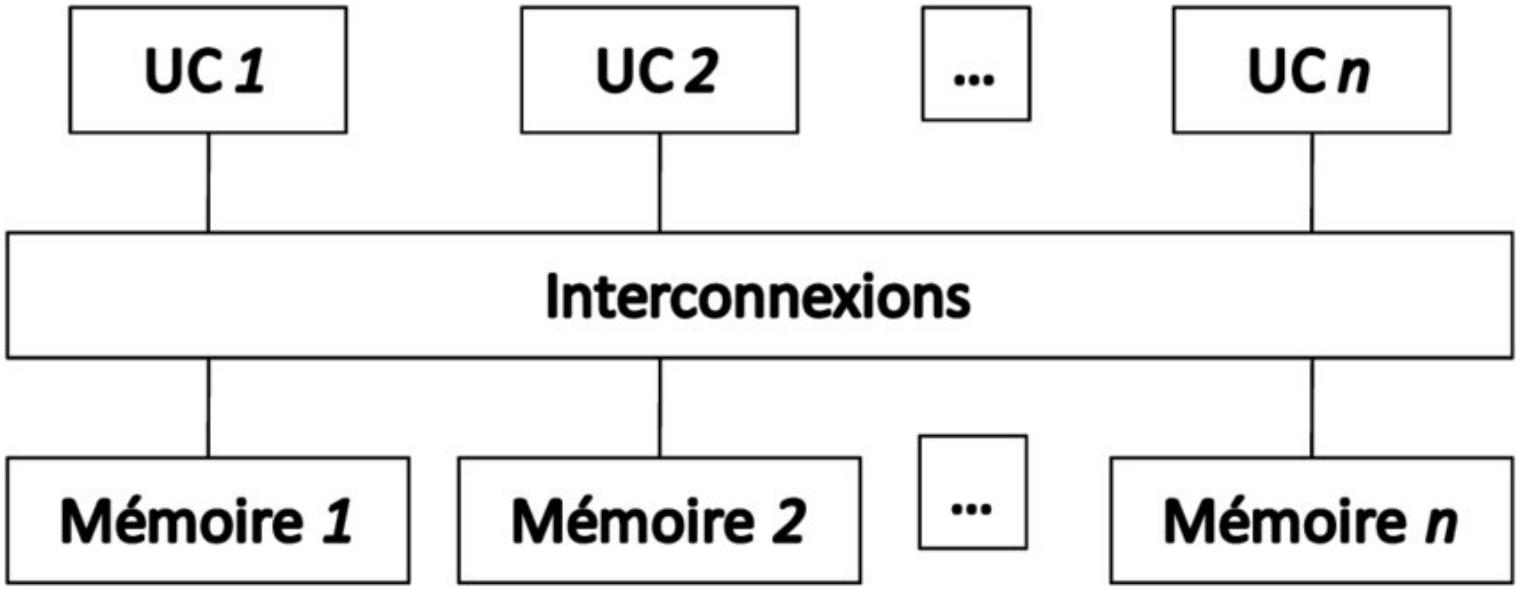
Protocole d’invalidation en écriture



Protocole d’invalidation en écriture



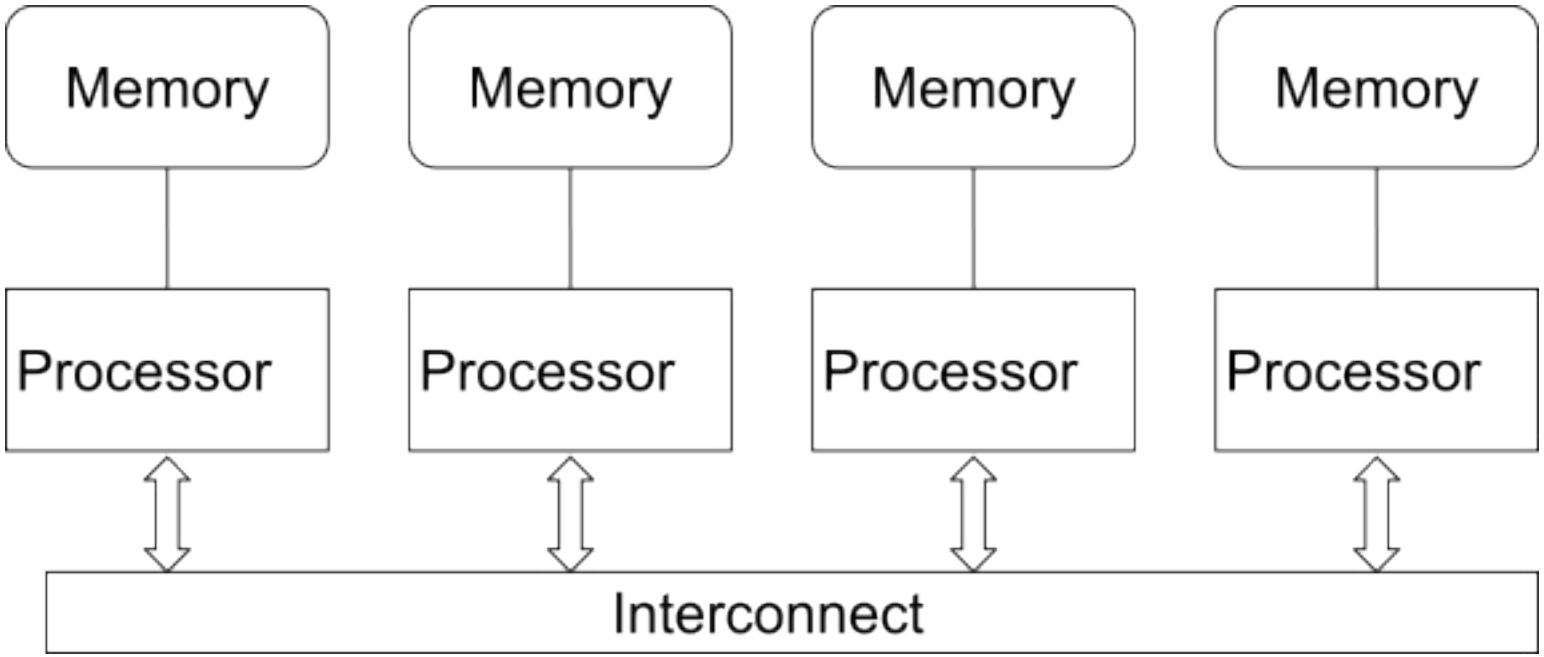
Accès uniforme à la mémoire (UMA) avec l'utilisation d'un réseau



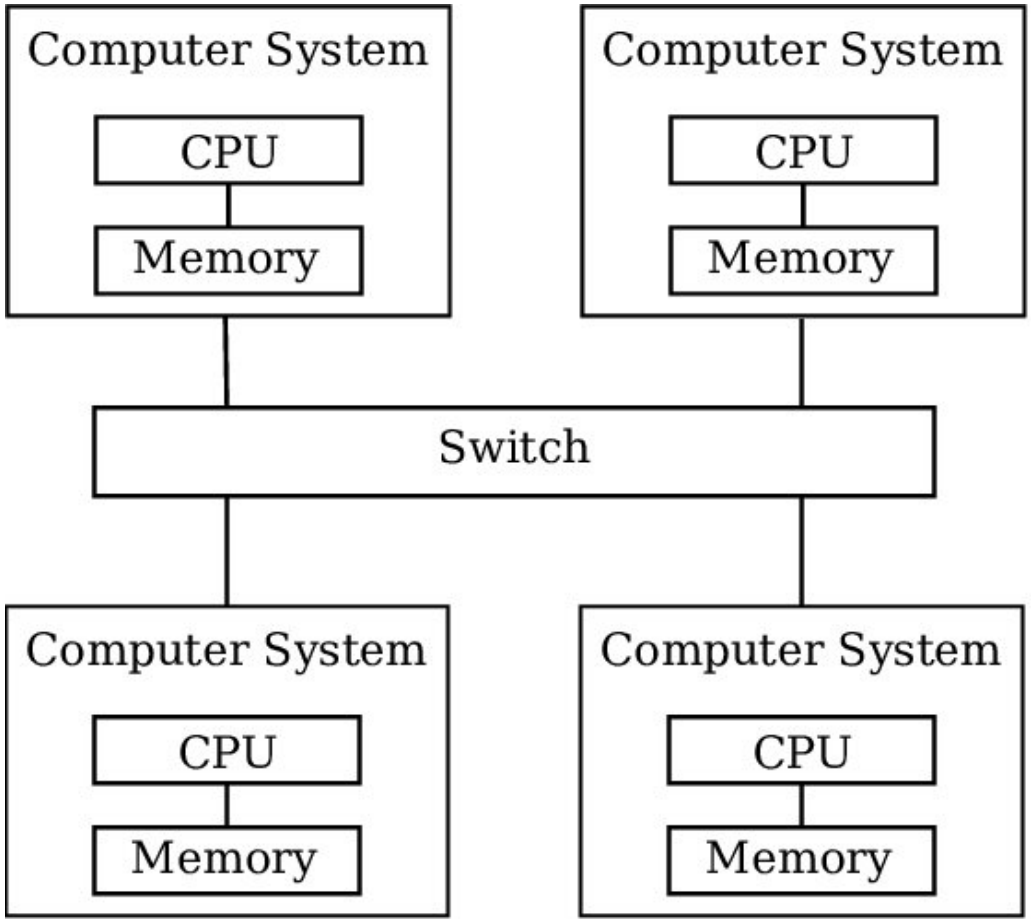
Multiprocesseurs avec mémoire distribuée (NUMA)

- La mémoire est distribuée entre les processeurs
- Diminue l’utilisation du réseau ainsi que le temps moyen d'accès à la mémoire.
- Permet d'utiliser un plus grand nombre de processeurs
- *Non-uniform memory access* (NUMA)

Multiprocesseurs distribués



Multiprocesseurs avec mémoire distribuée



Cohérence de la mémoire cache

- Implémentation plus difficile que pour les multiprocesseurs à mémoire partagées.
  - Pas de bus unique à monitorer
  - Méthode la plus utilisée: *protocole à répertoire* (Directory-based protocol)

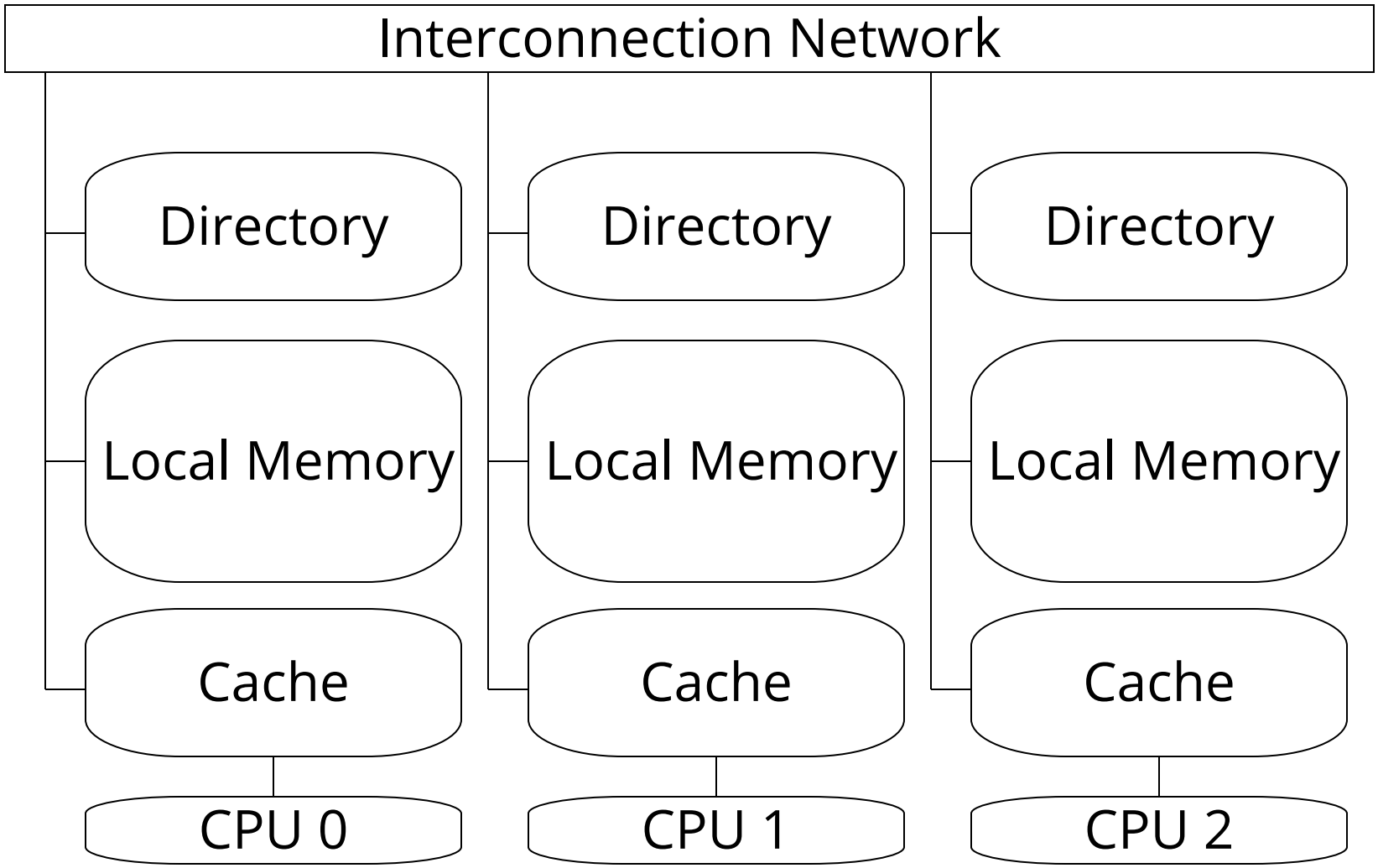
Protocole à répertoire

- Un répertoire contient l'information concernant les blocs de mémoire pouvant être mis en cache
- Le répertoire est distribué
- Une entrée dans le répertoire pour chaque bloc de mémoire
- Chaque entrée possède:
  - Status de partage
  - Liste des processeurs possédant une copie du bloc

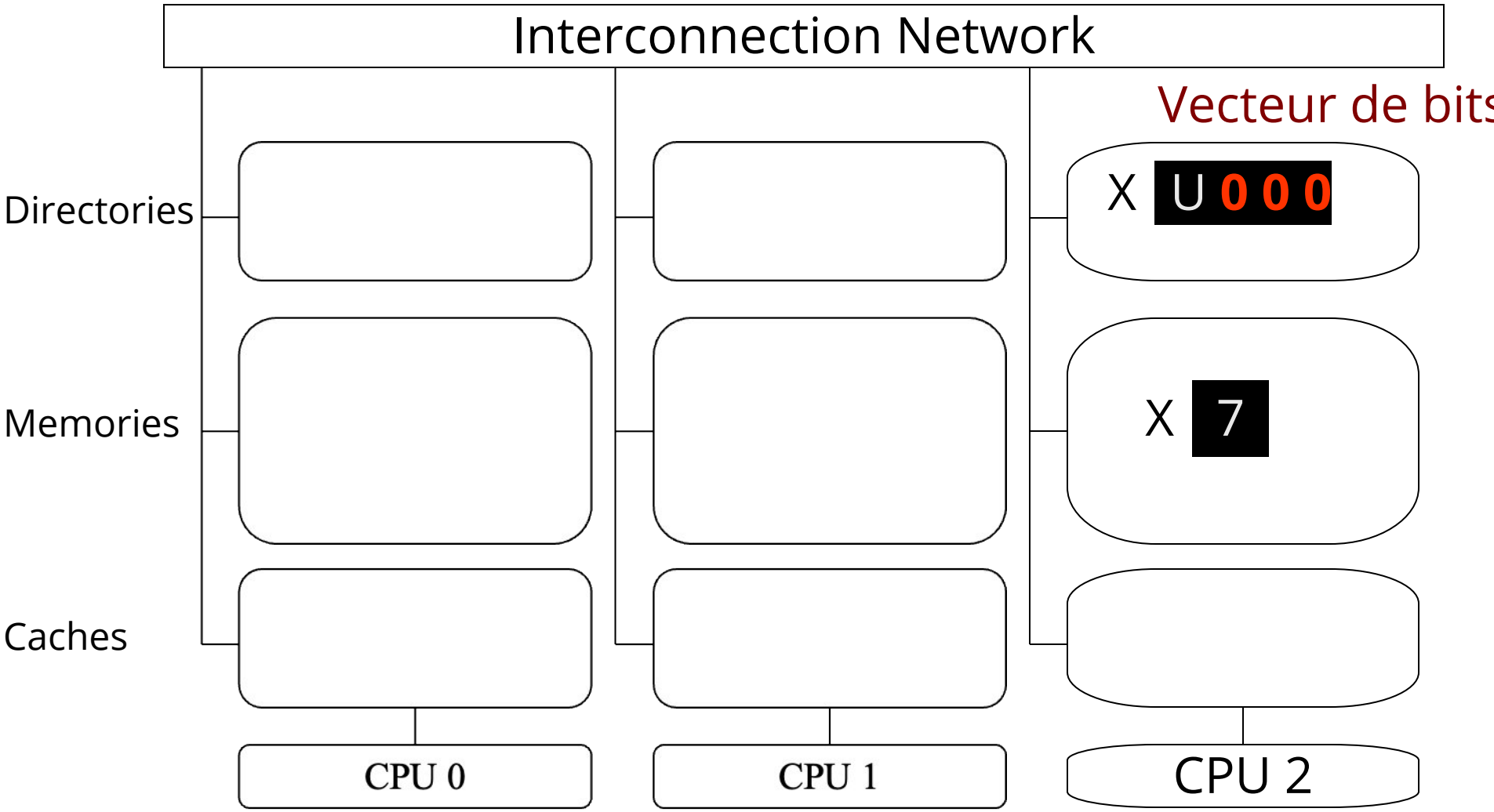
Status de partage

- Pas en cache (U)
  - Aucun processeur n’a mis le bloc en cache
- Partagé (S)
  - Dans le cache d’un ou plusieurs processeurs
  - En lecture seulement
- Exclusif (E)
  - Dans le cache d’un seule processeur
  - Le bloc a été modifié
  - La copie en mémoire n’est plus valide

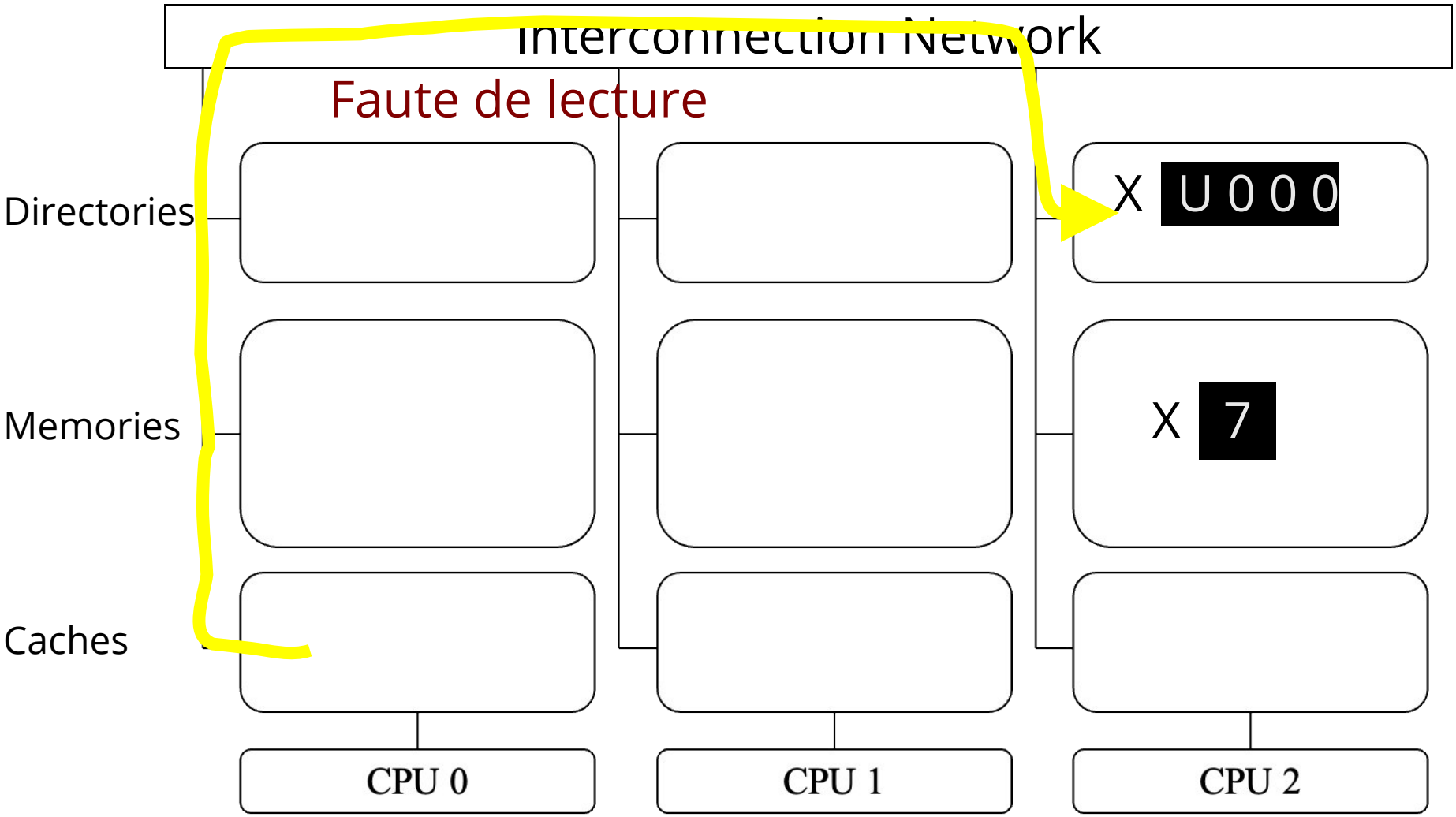
Protocole avec répertoire



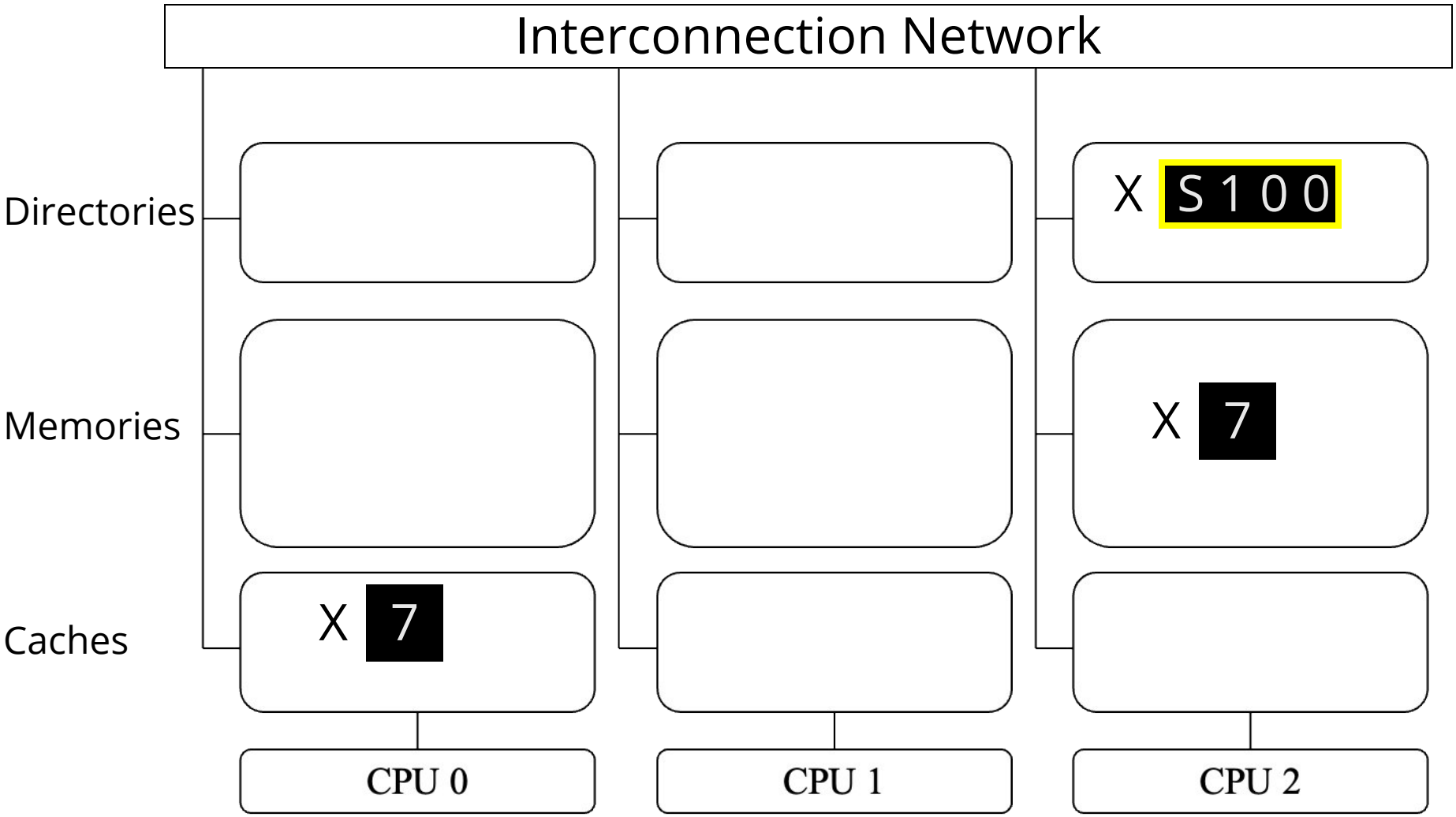
Protocole avec répertoire



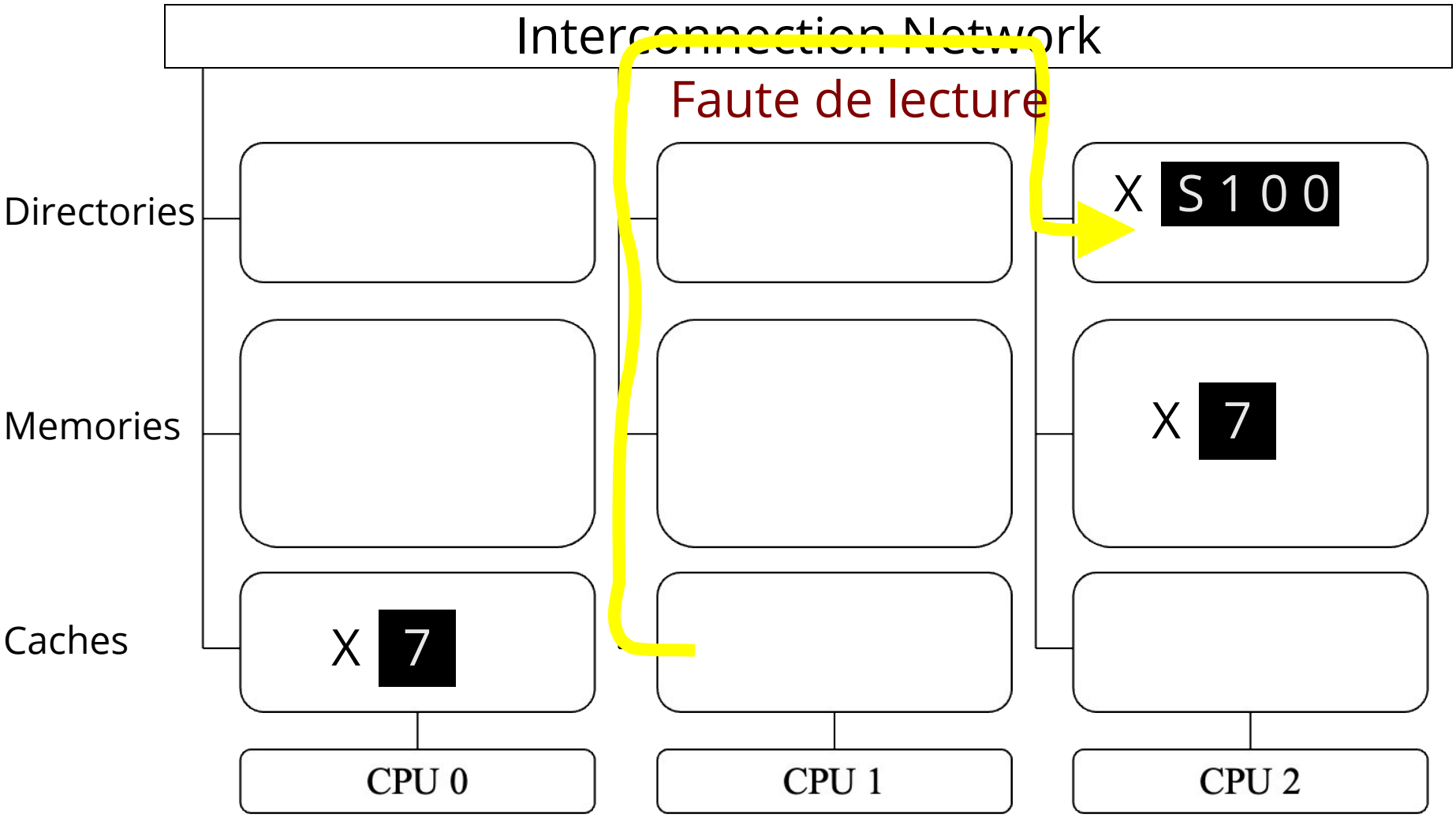
Processeur 0 lit X



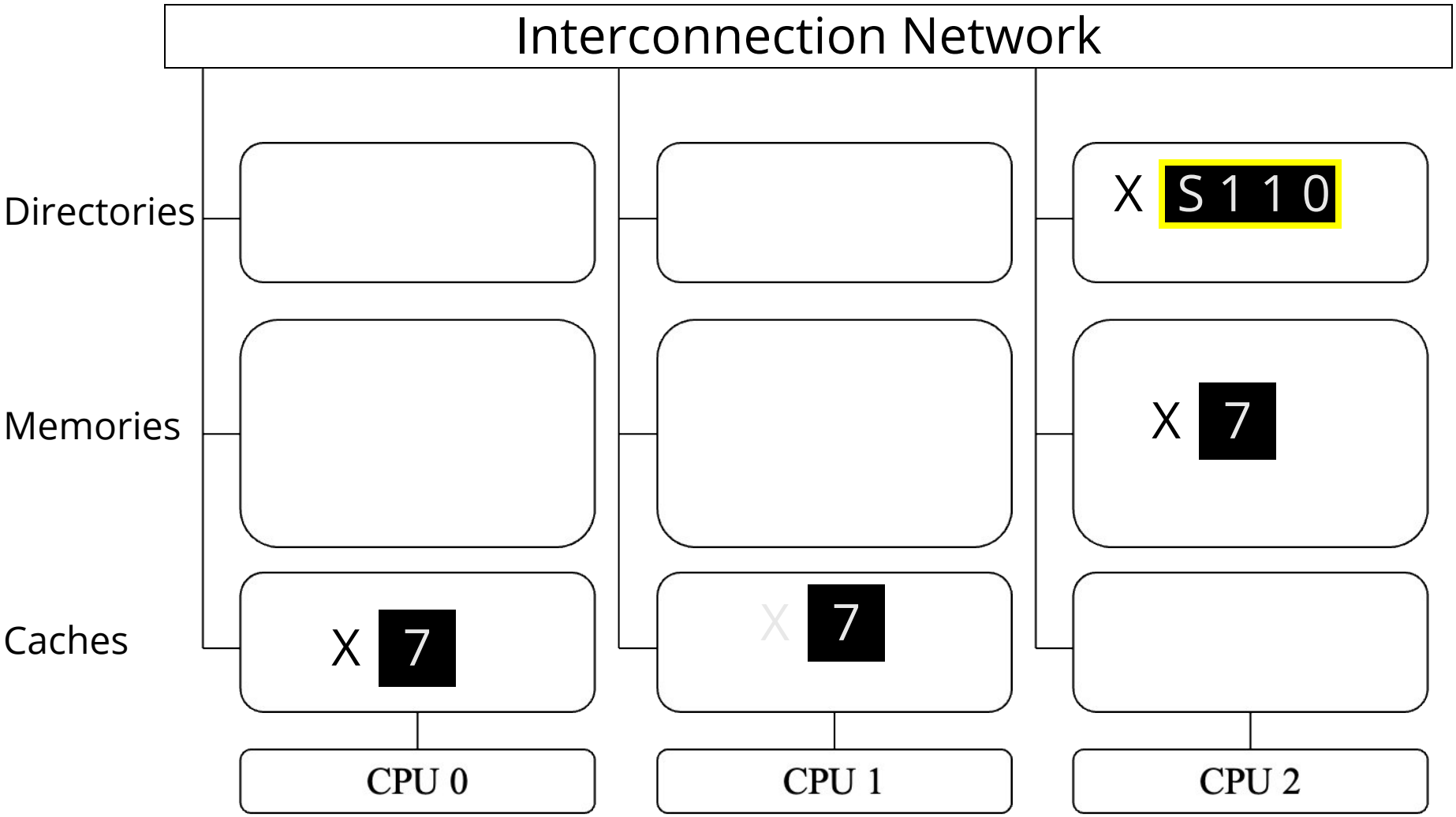
Processeur 0 lit X



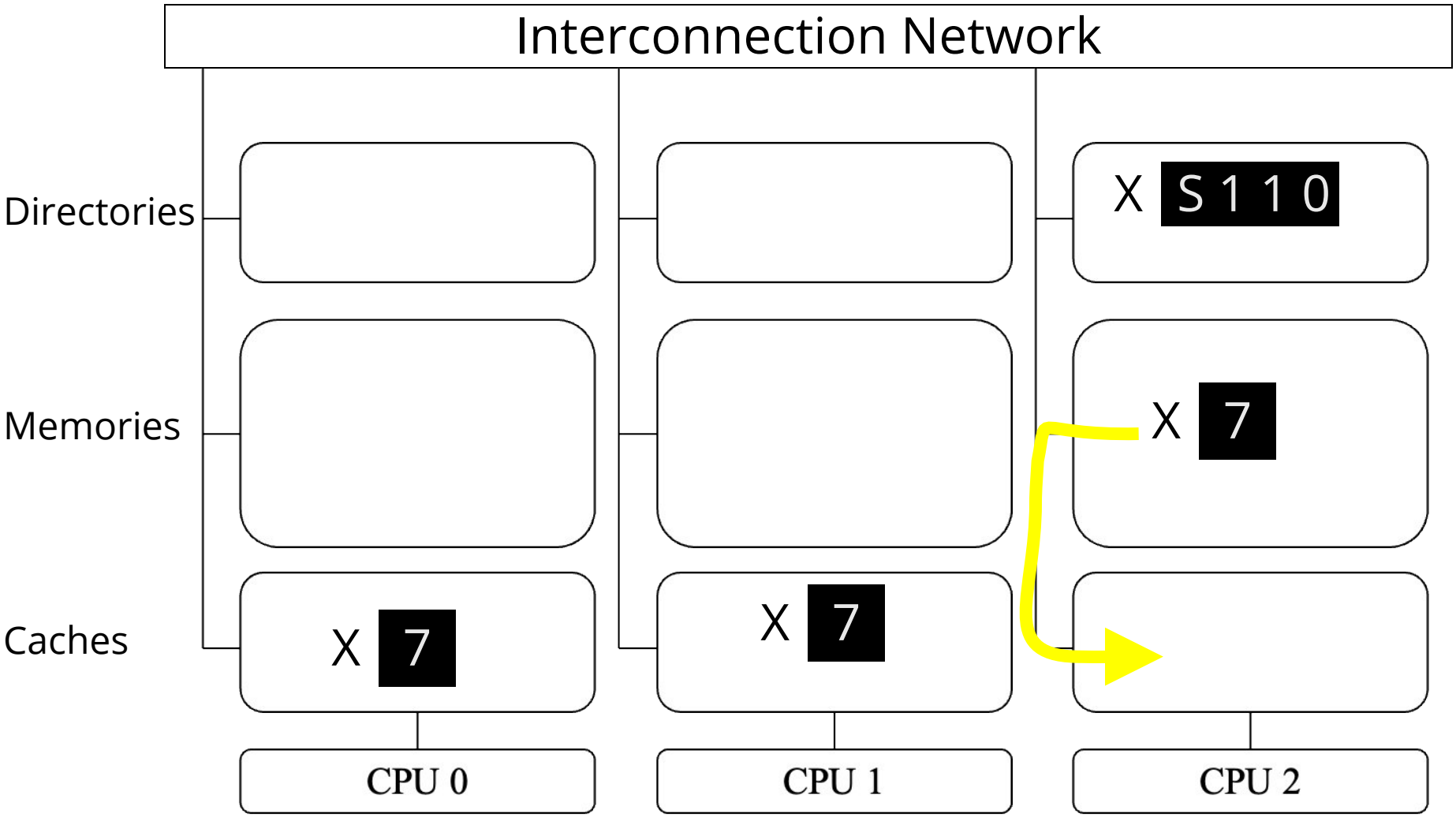
Processeur 1 lit X



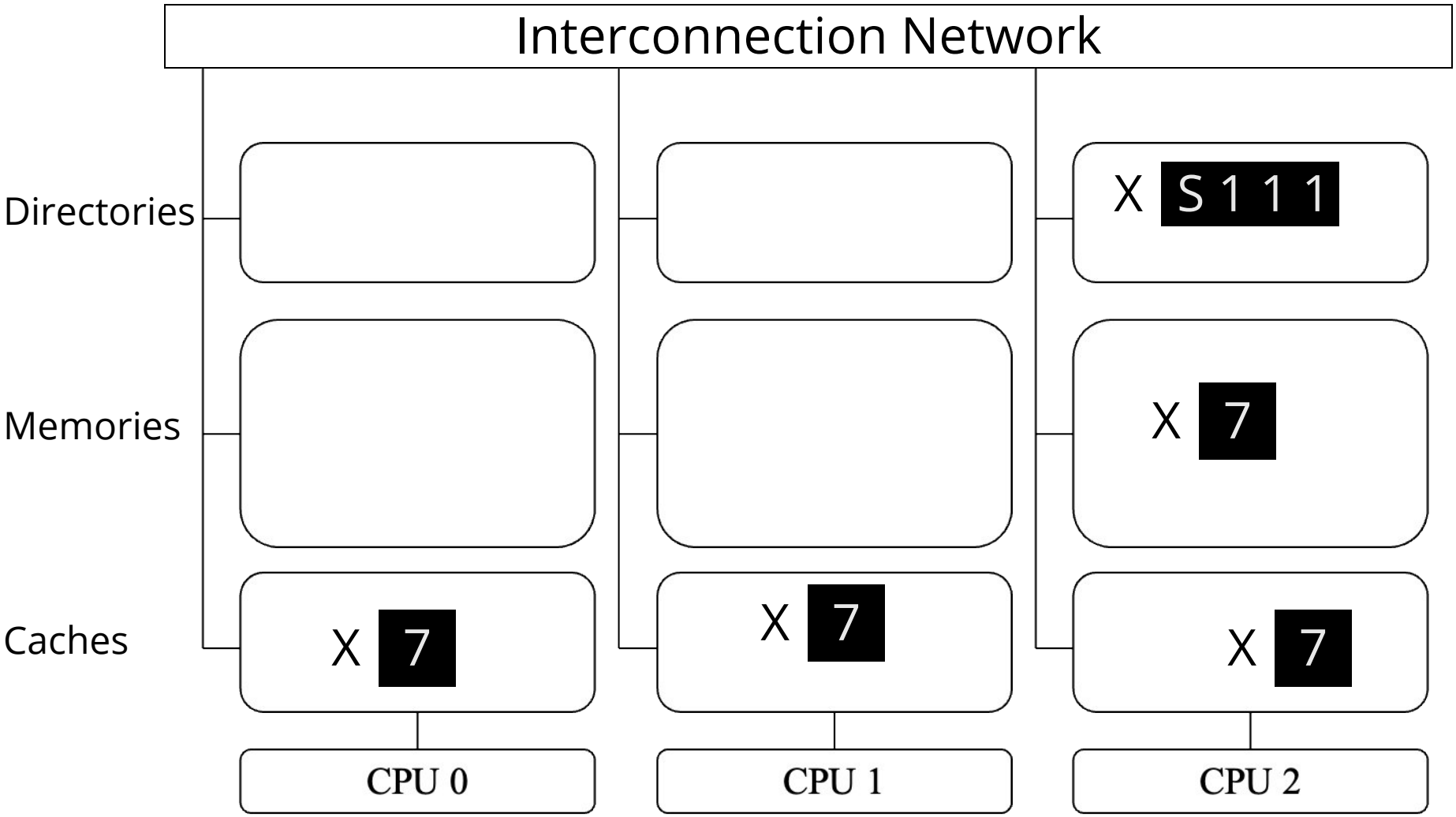
Processeur 1 lit X



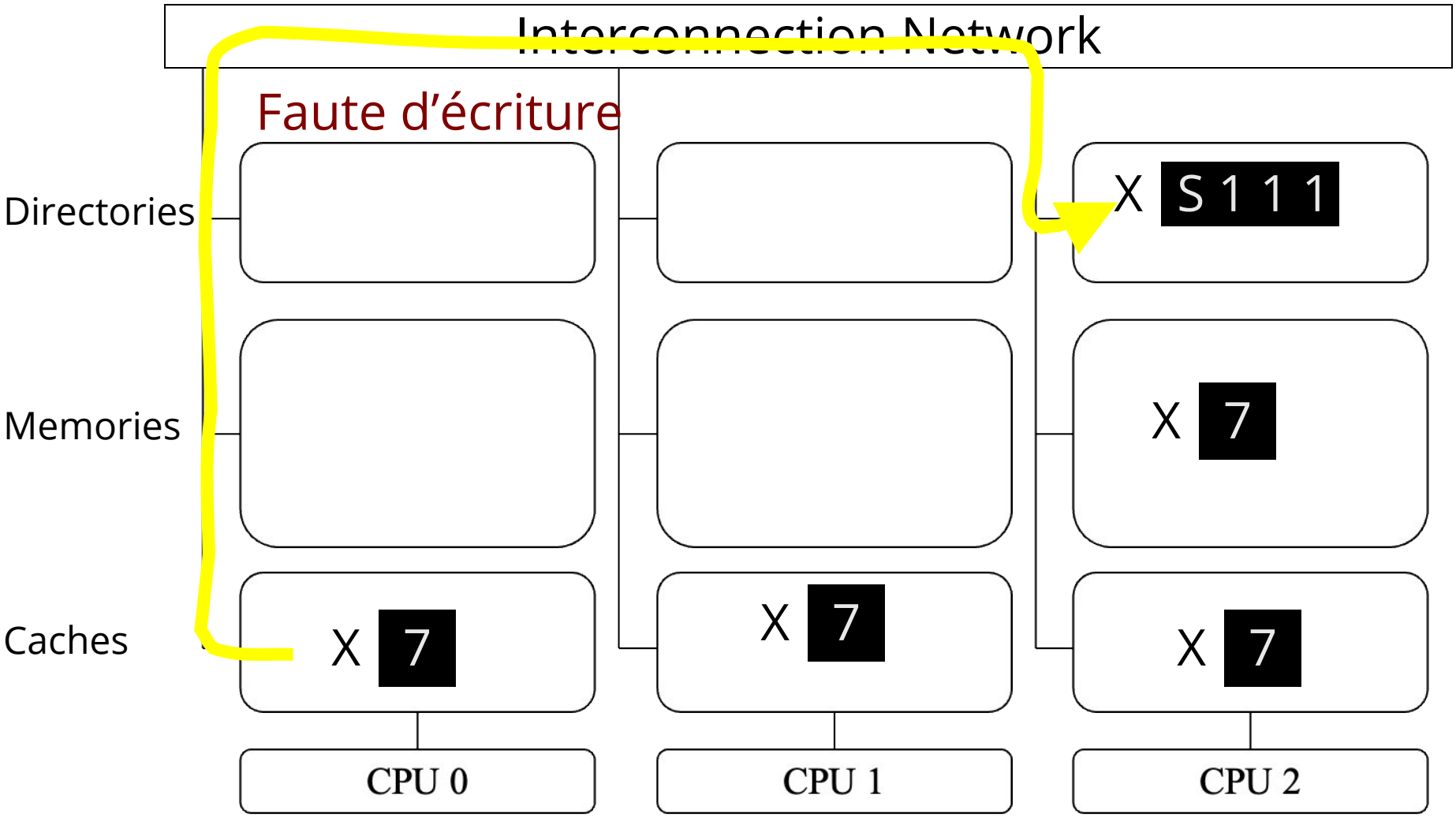
Processeur 2 lit X



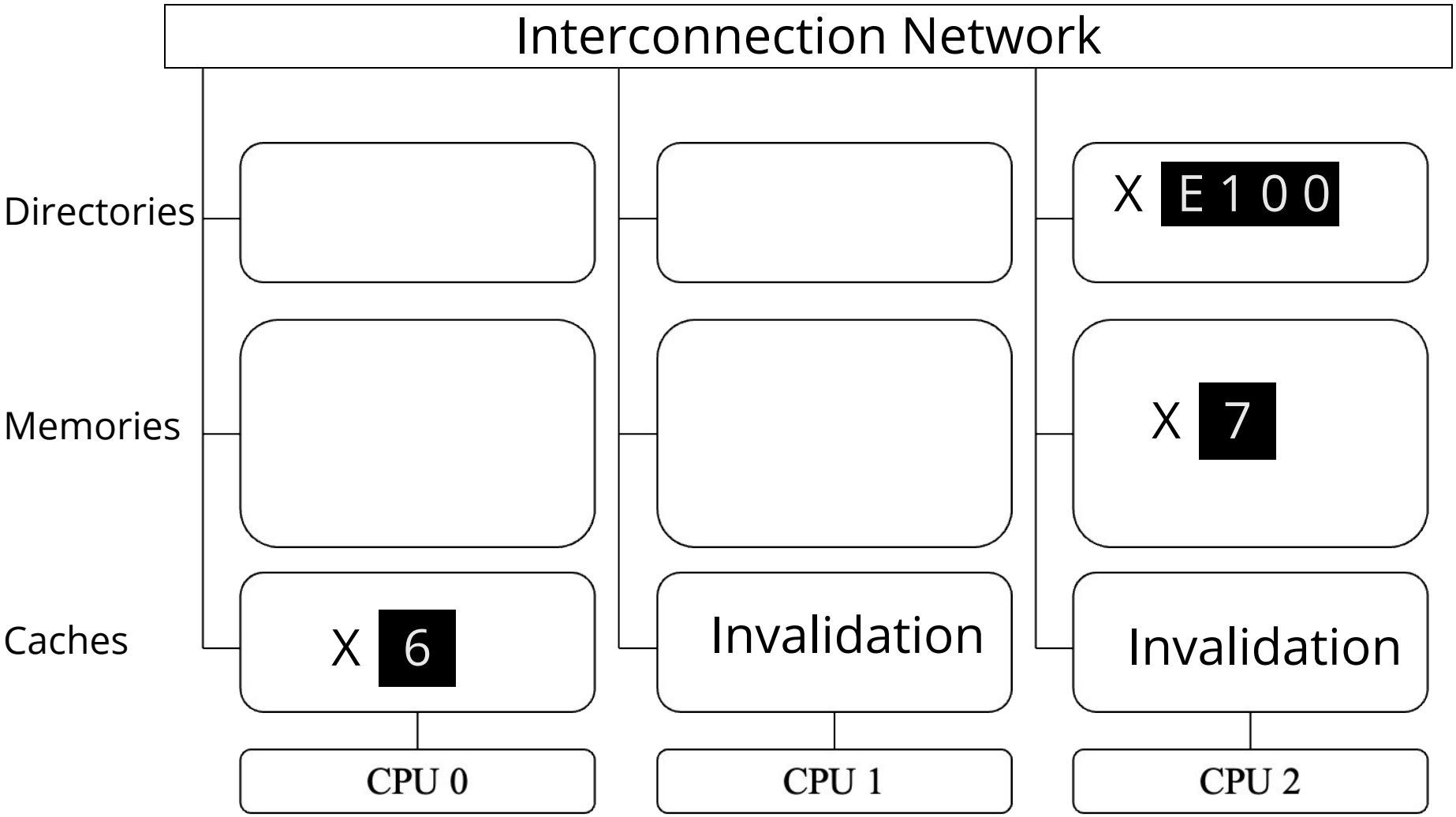
Processeur 2 lit X



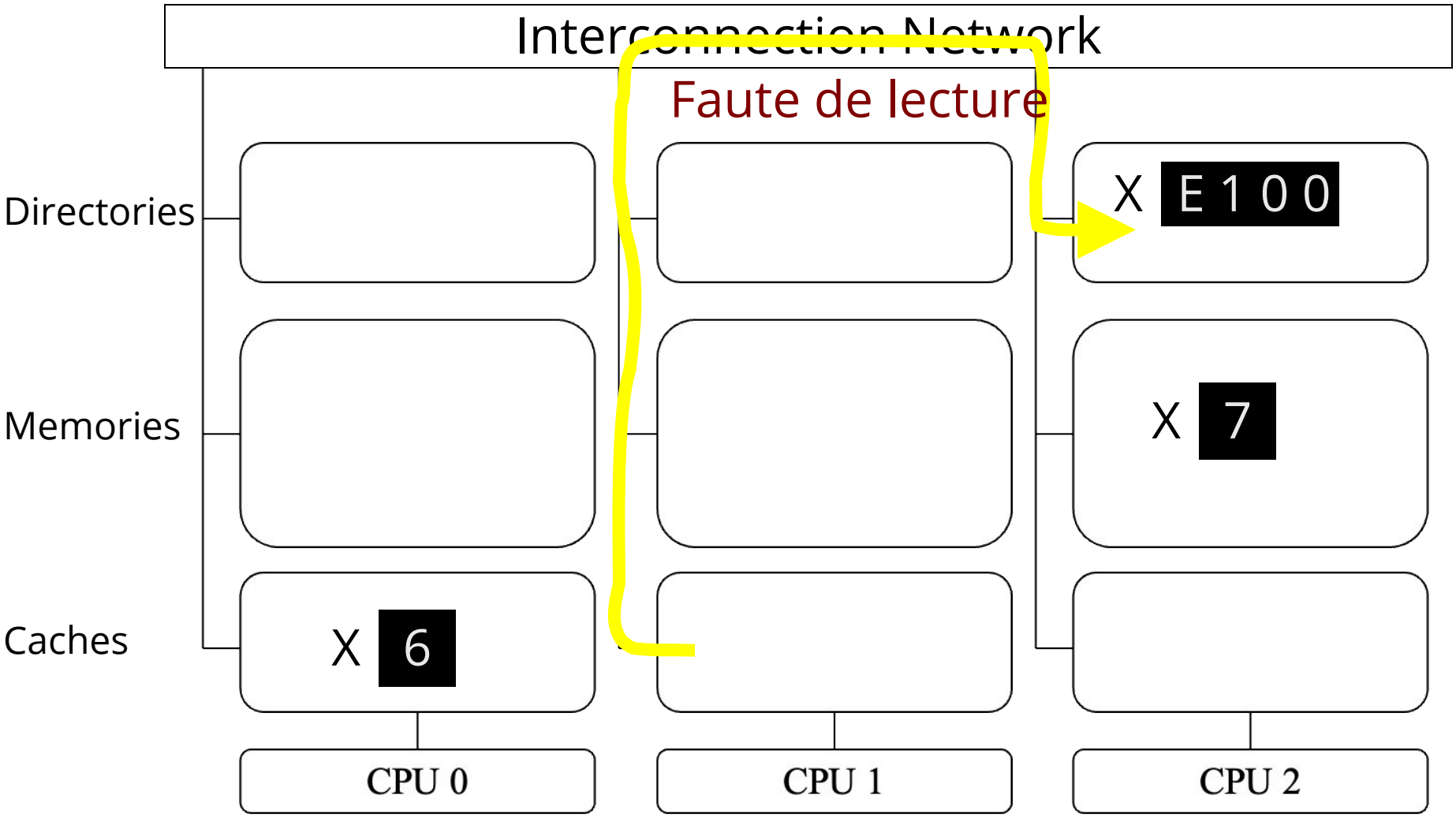
Processeur 0 écrit 6 dans X



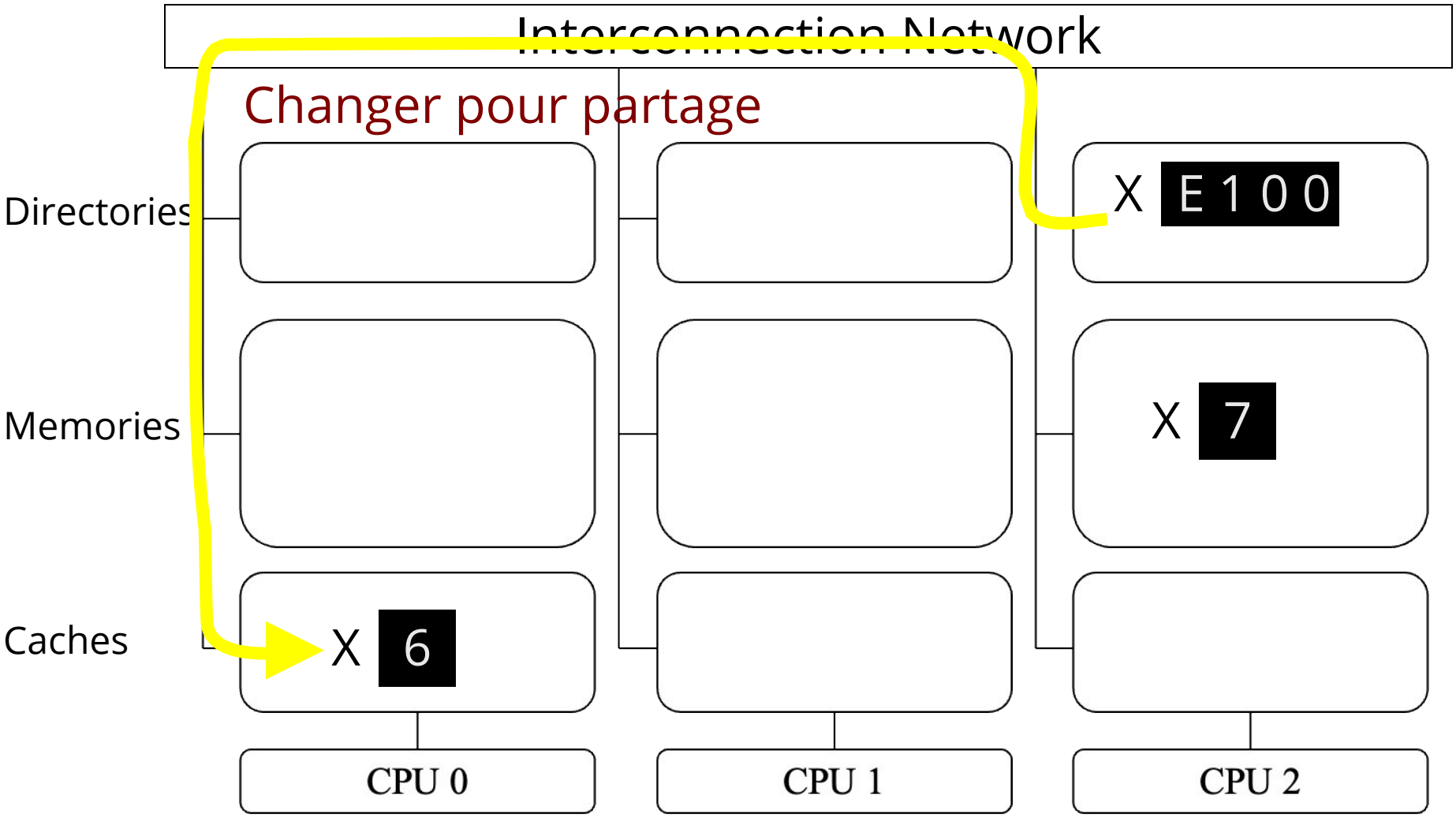
Processeur 0 écrit 6 dans X



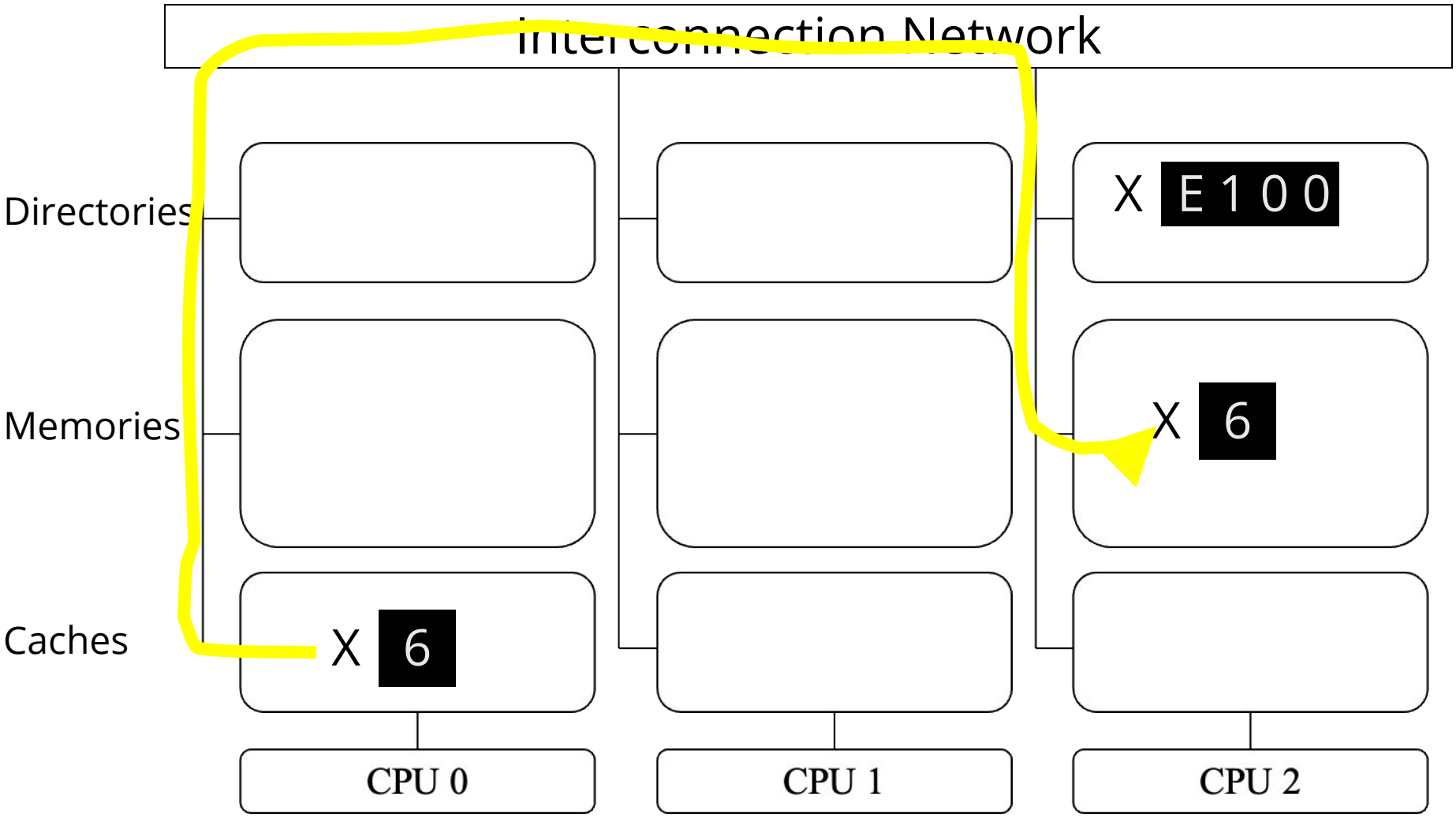
Processeur 1 lit X



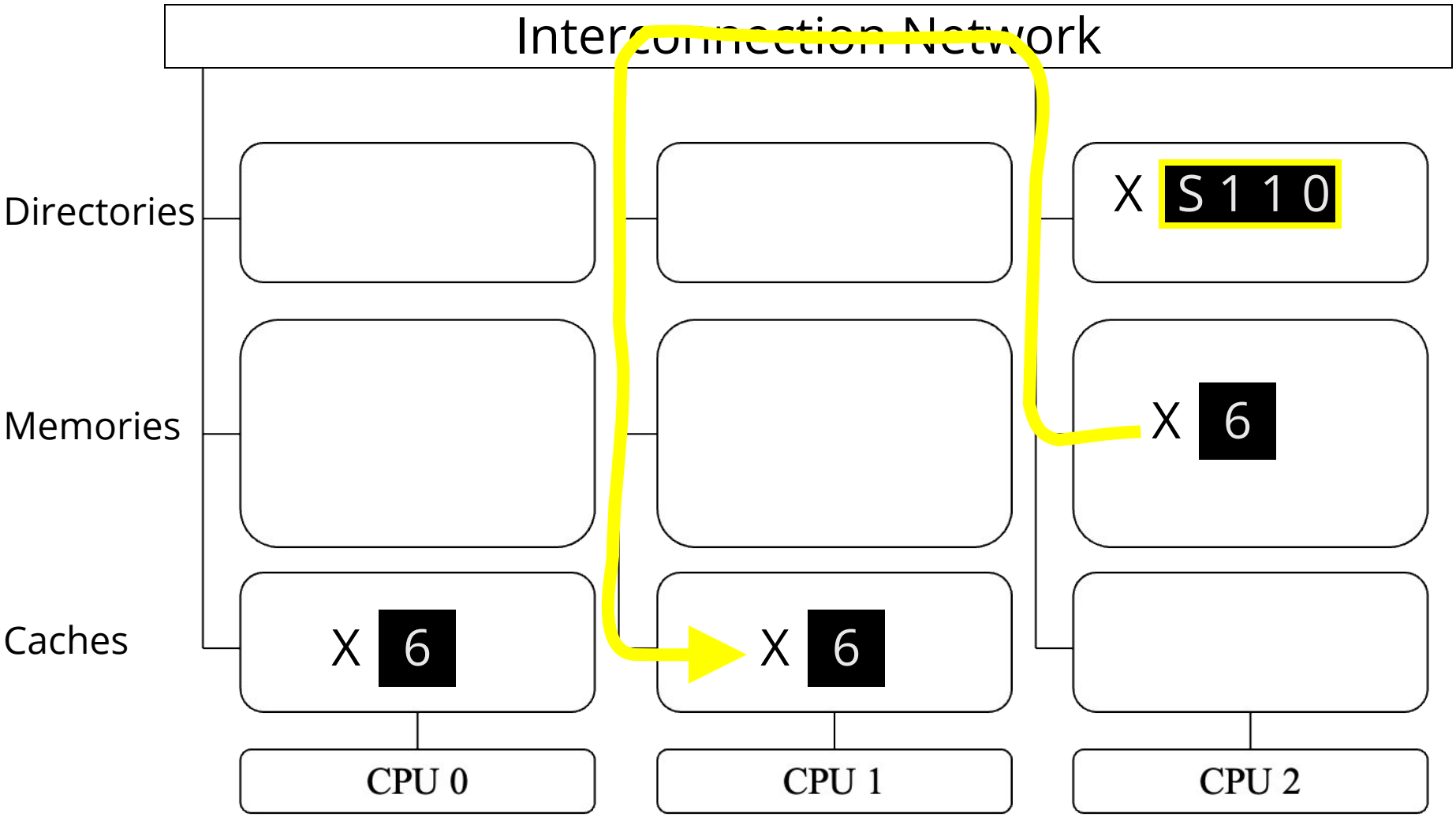
Processeur 1 lit X



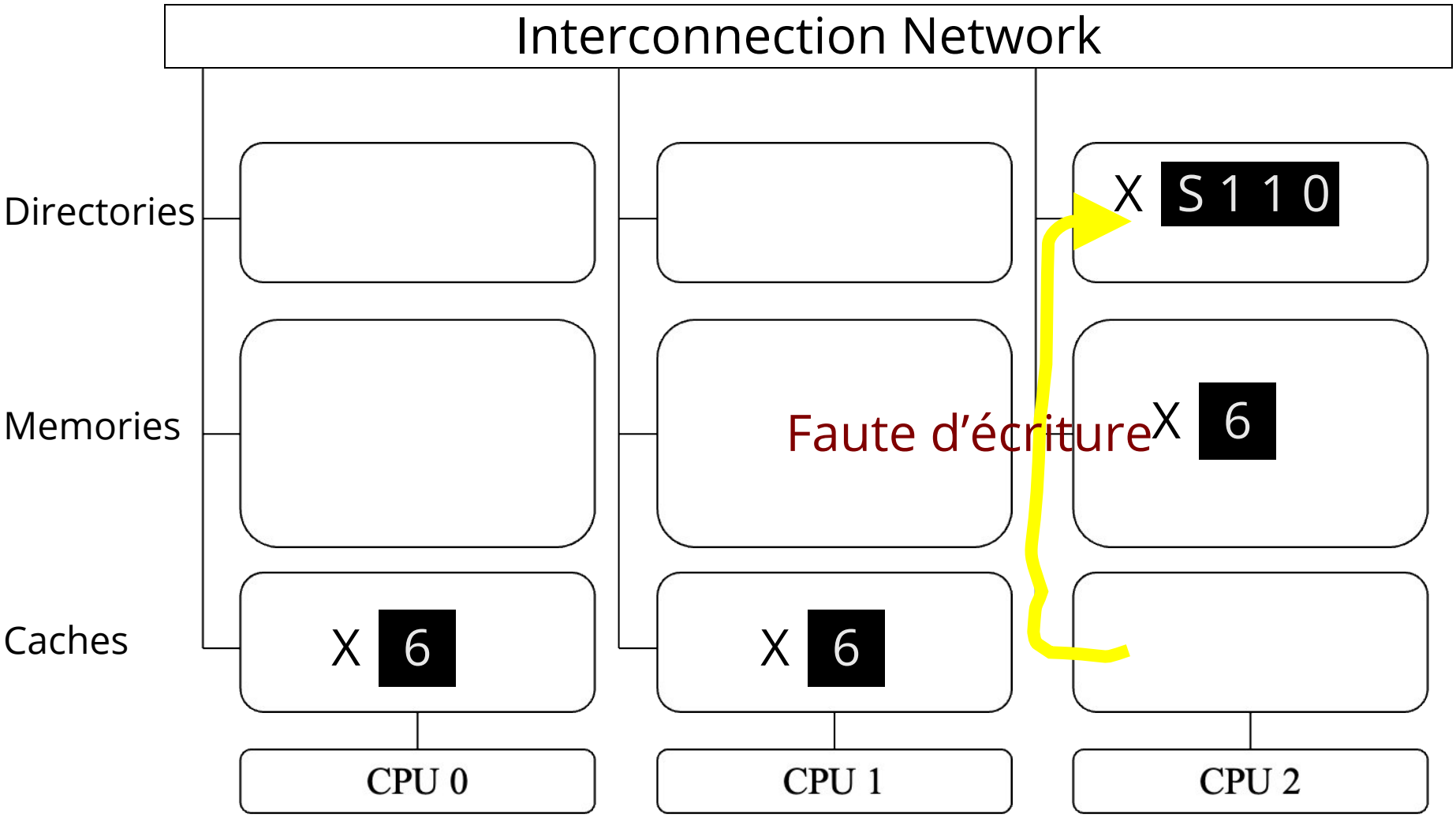
Processeur 1 lit X



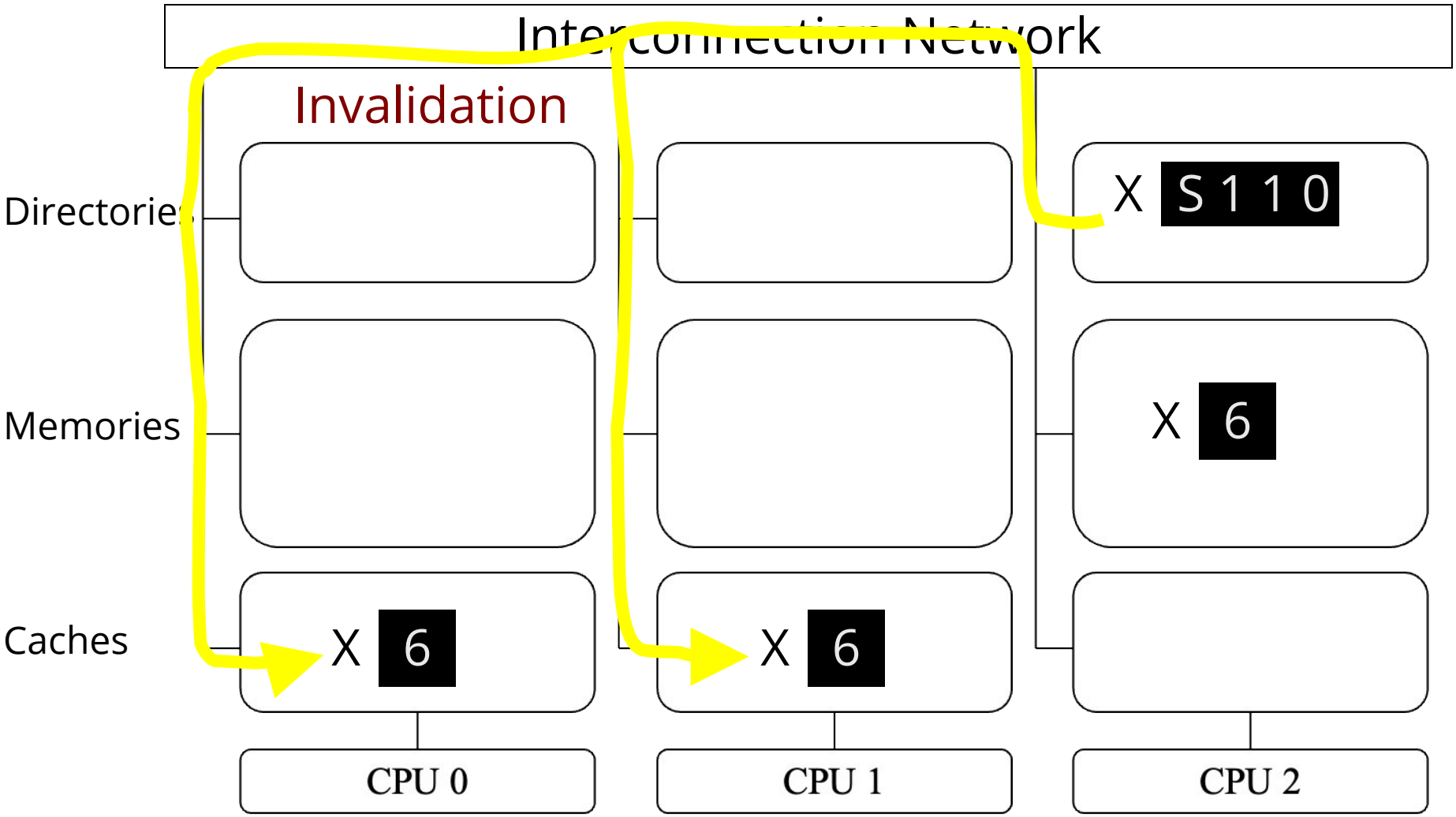
Processeur 1 lit X



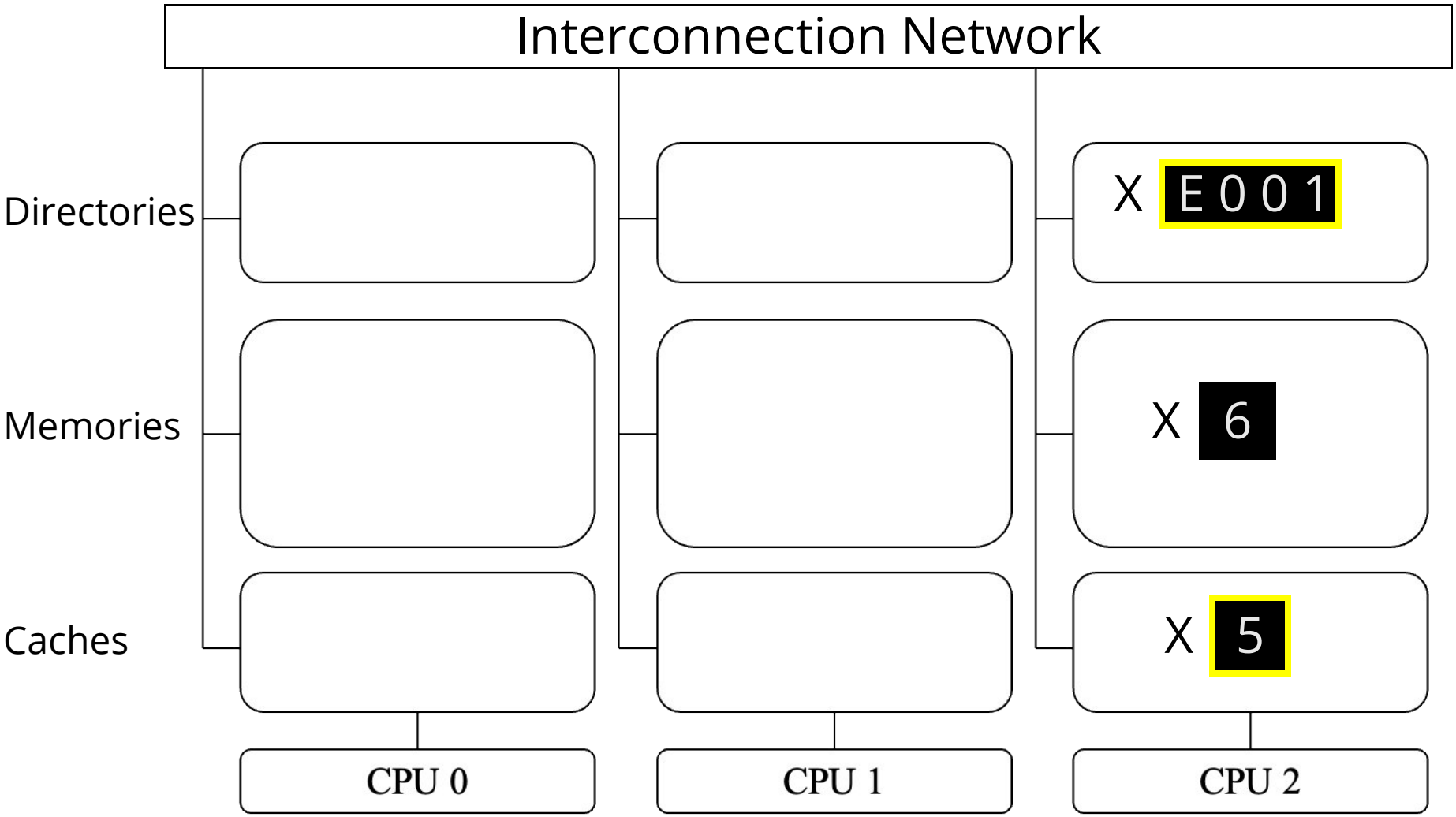
Processeur 2 met 5 dans X



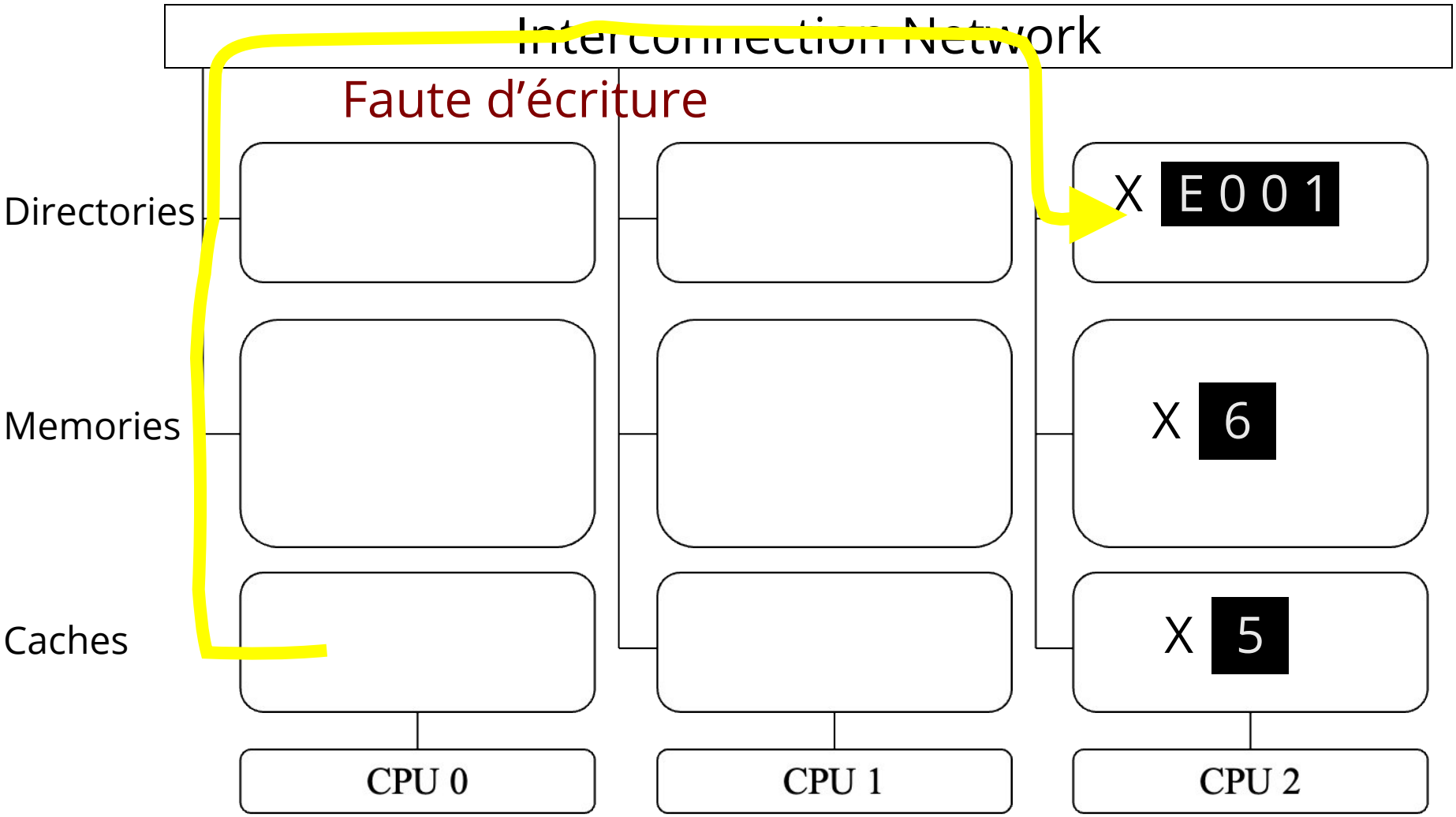
Processeur 2 met 5 dans X



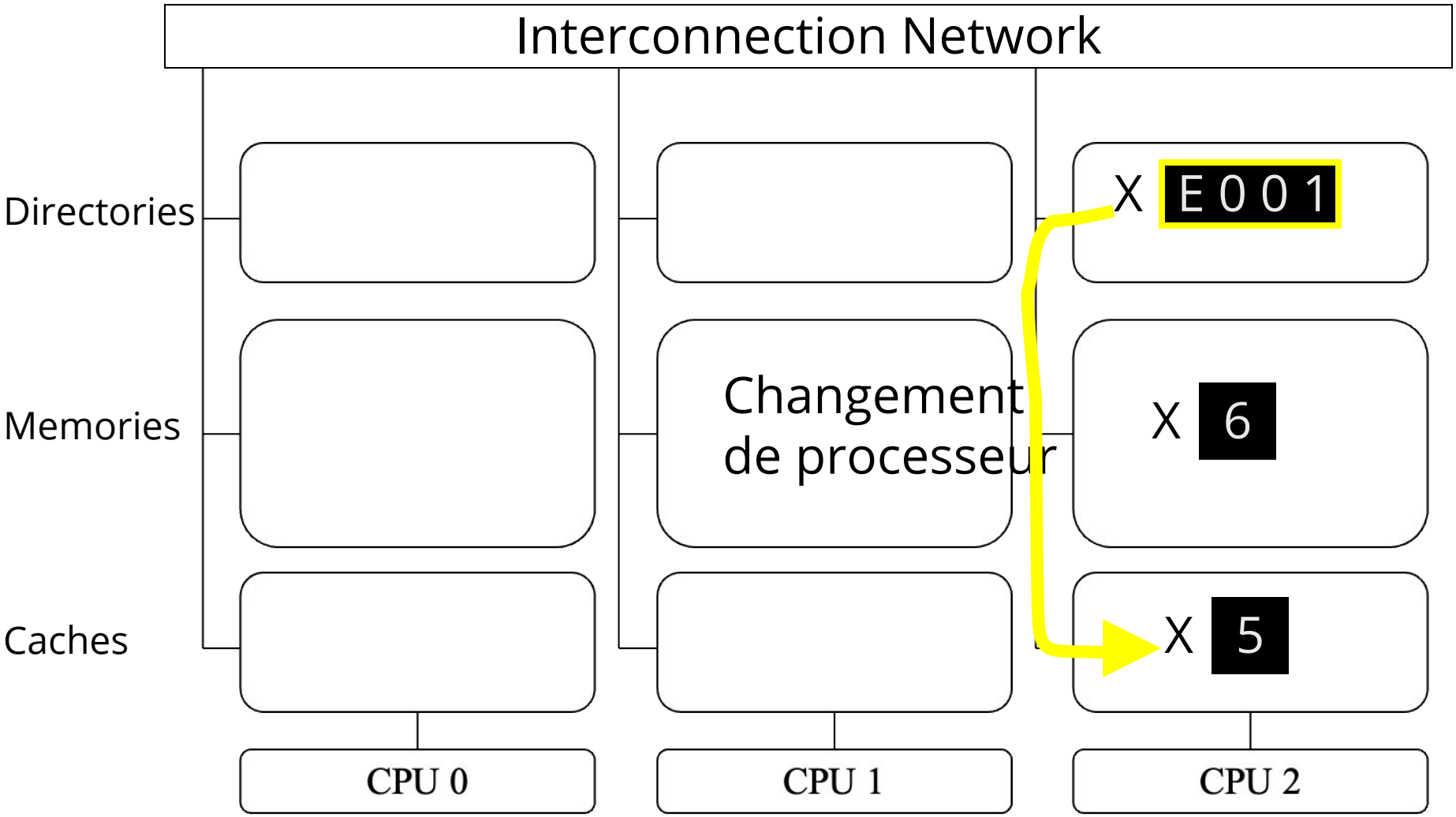
Processeur 2 met 5 dans X



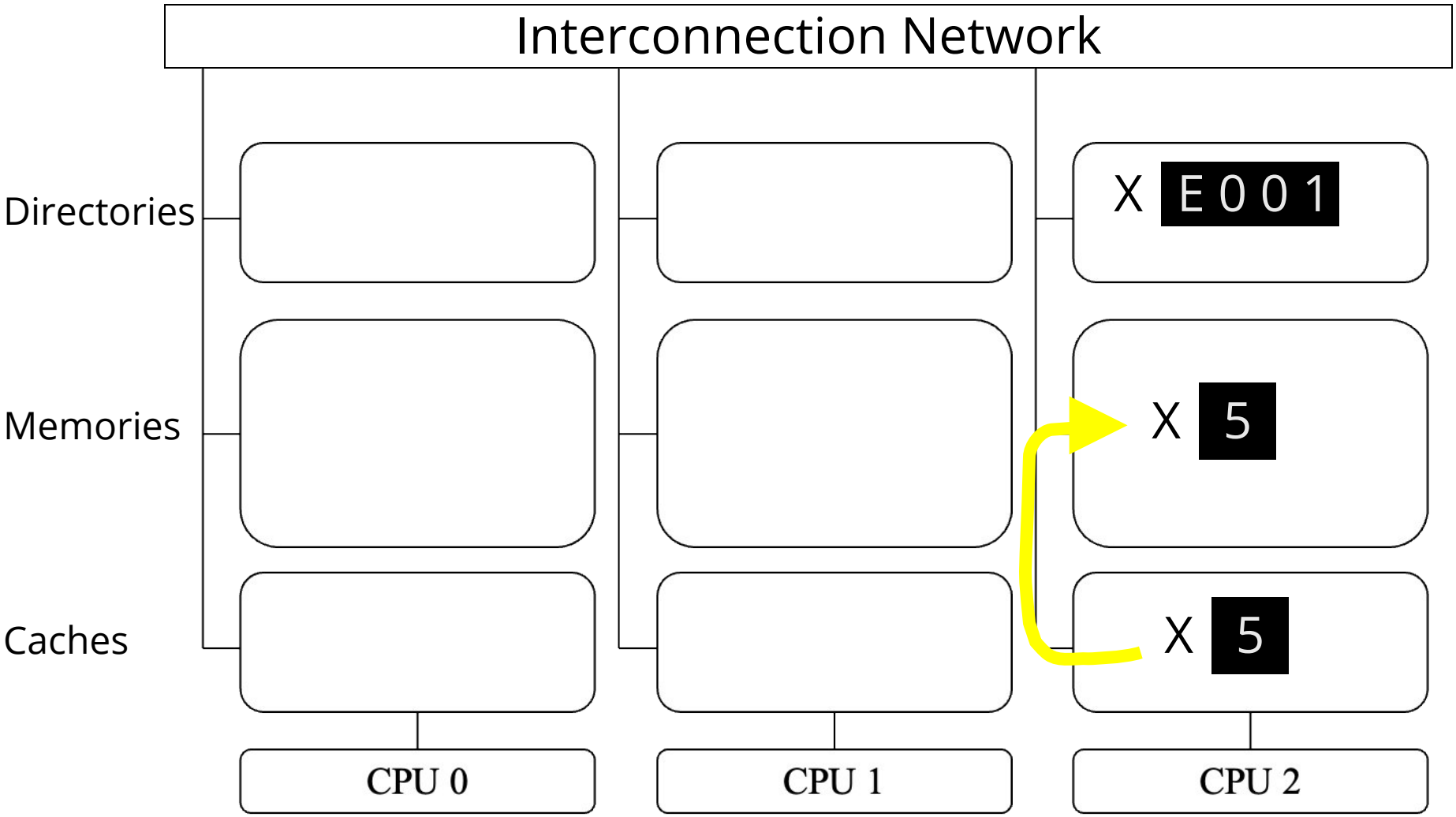
Processeur 0 met 4 dans X



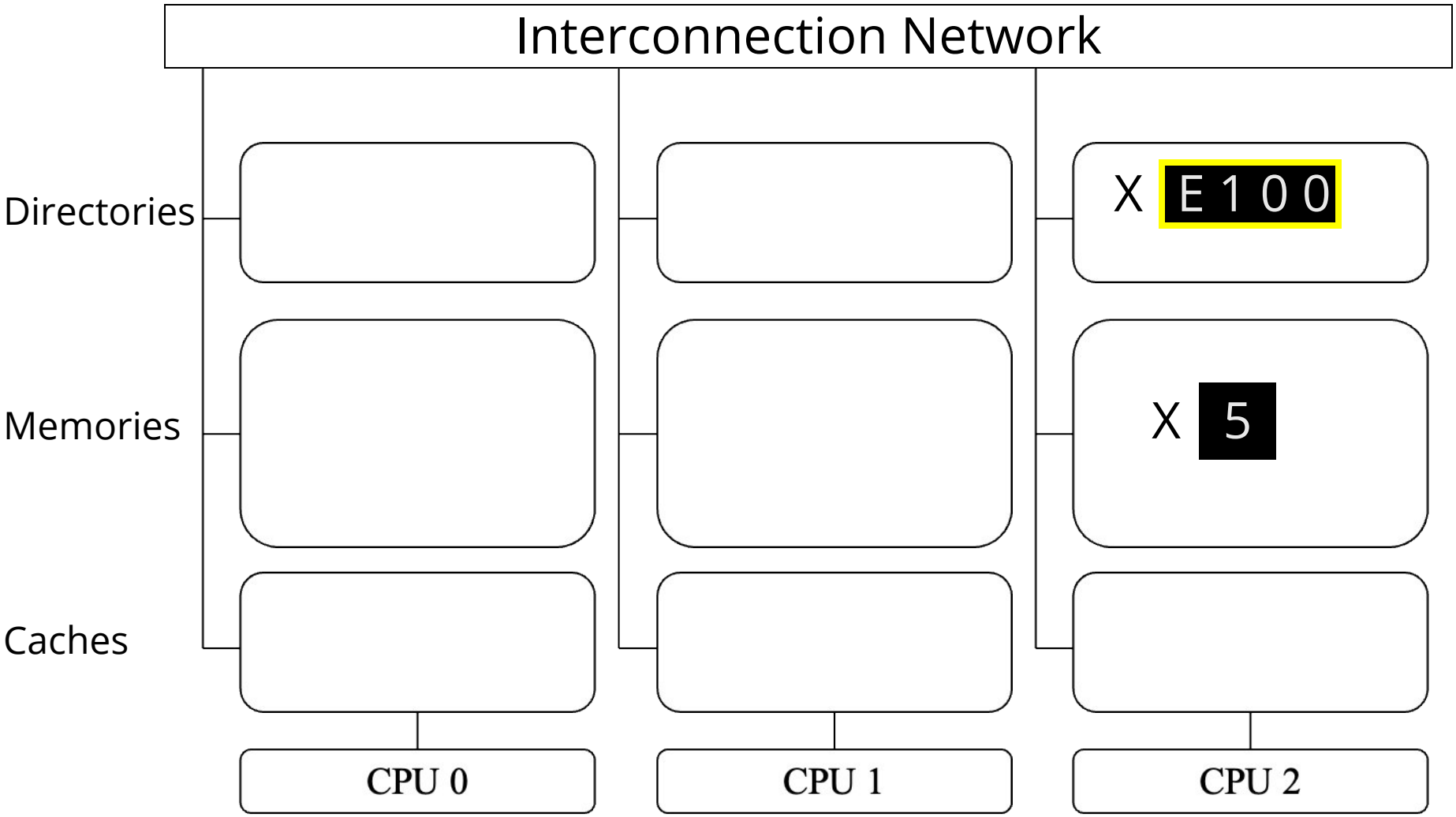
Processeur 0 met 4 dans X



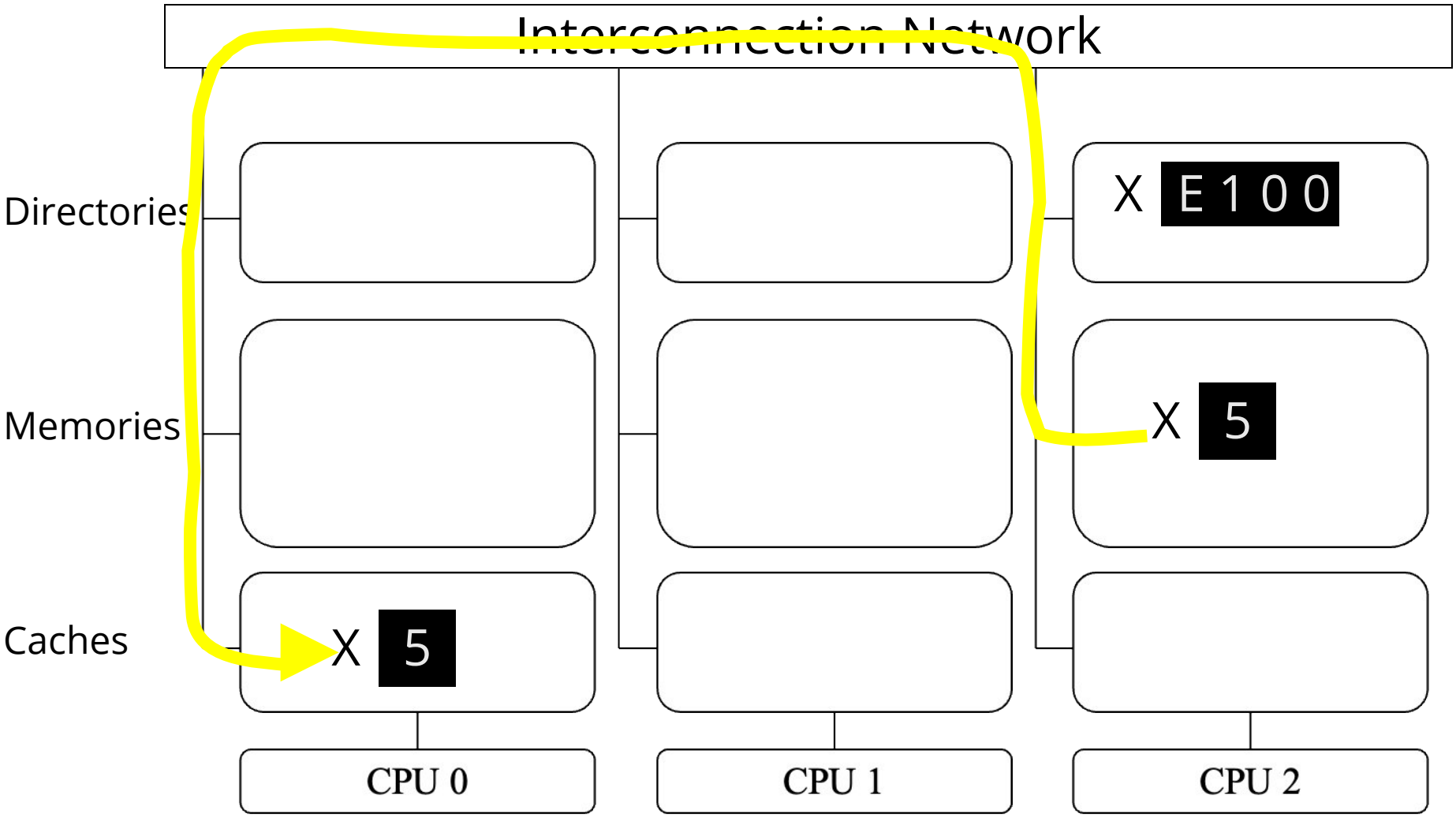
Processeur 0 met 4 dans X



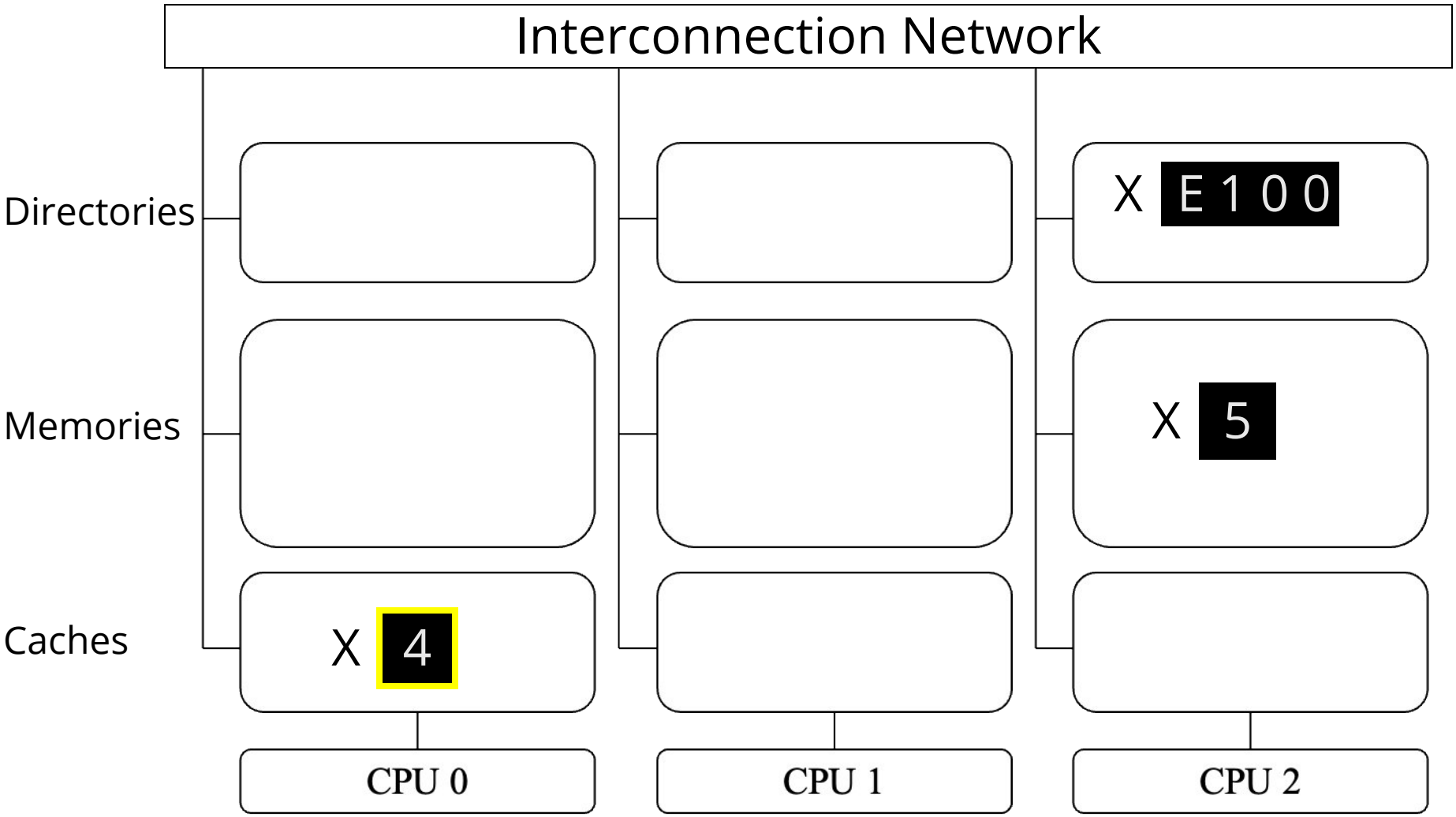
Processeur 0 met 4 dans X



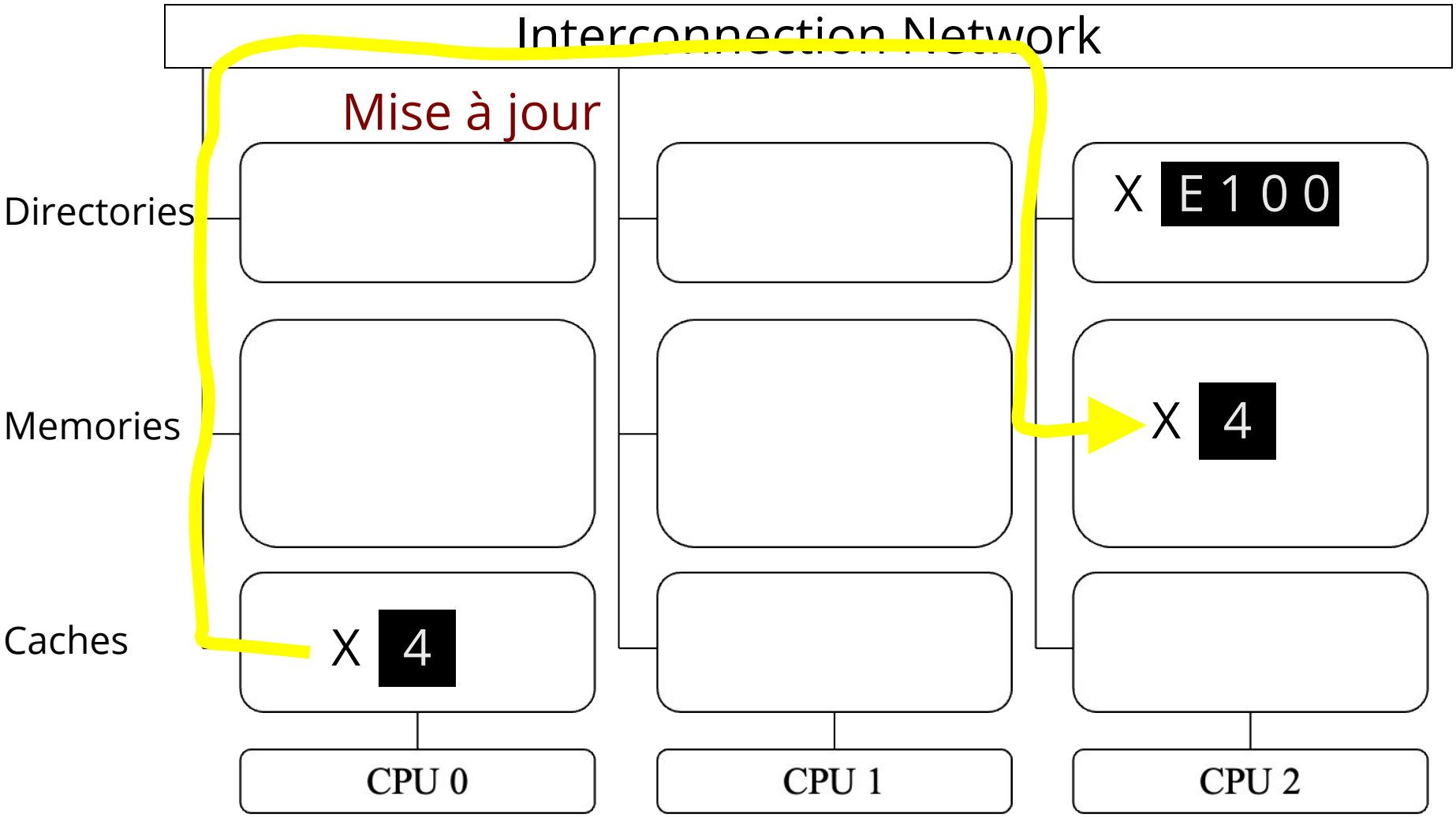
Processeur 0 met 4 dans X



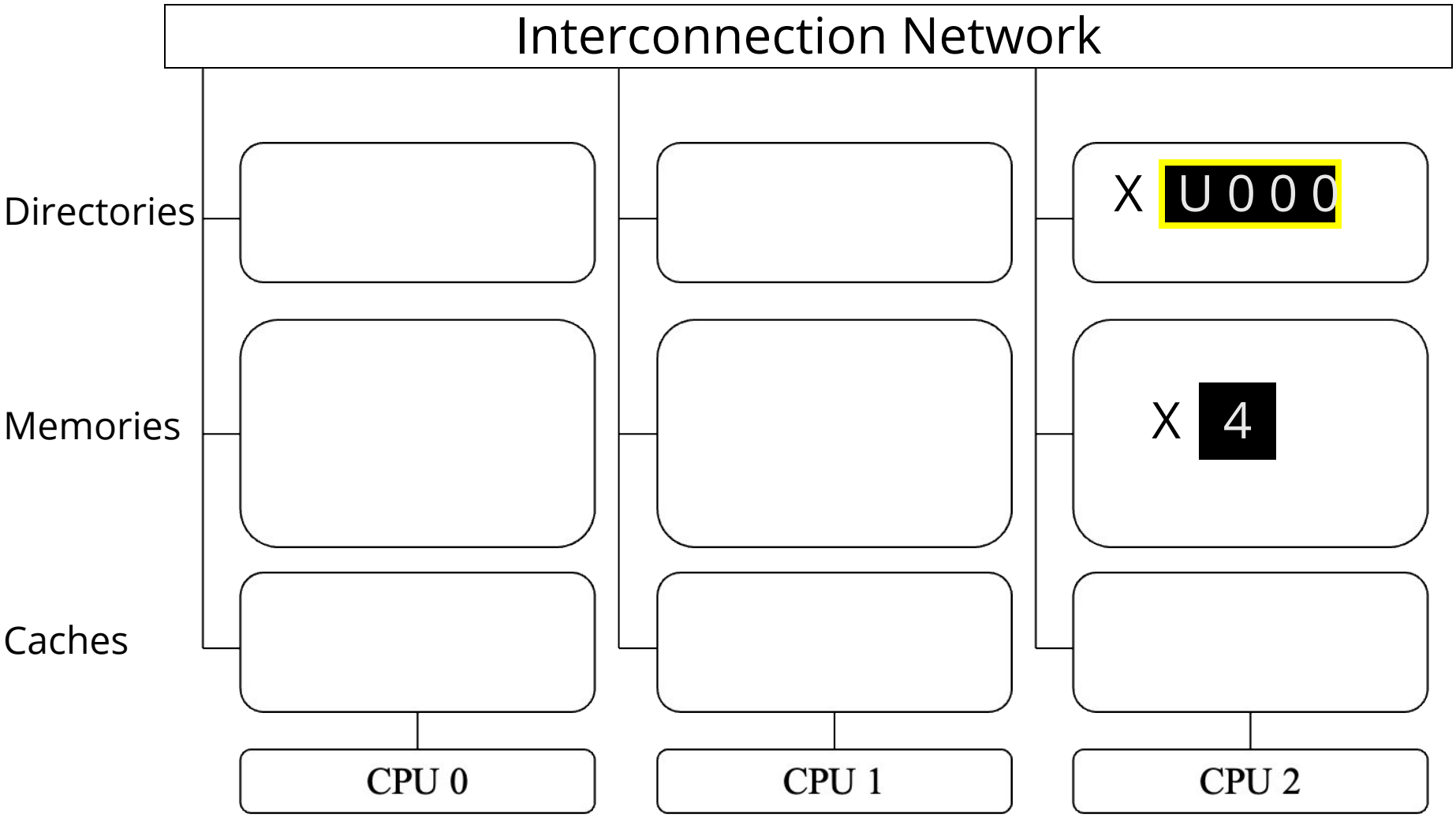
Processeur 0 met 4 dans X



Processeur 0 libère X (flush)



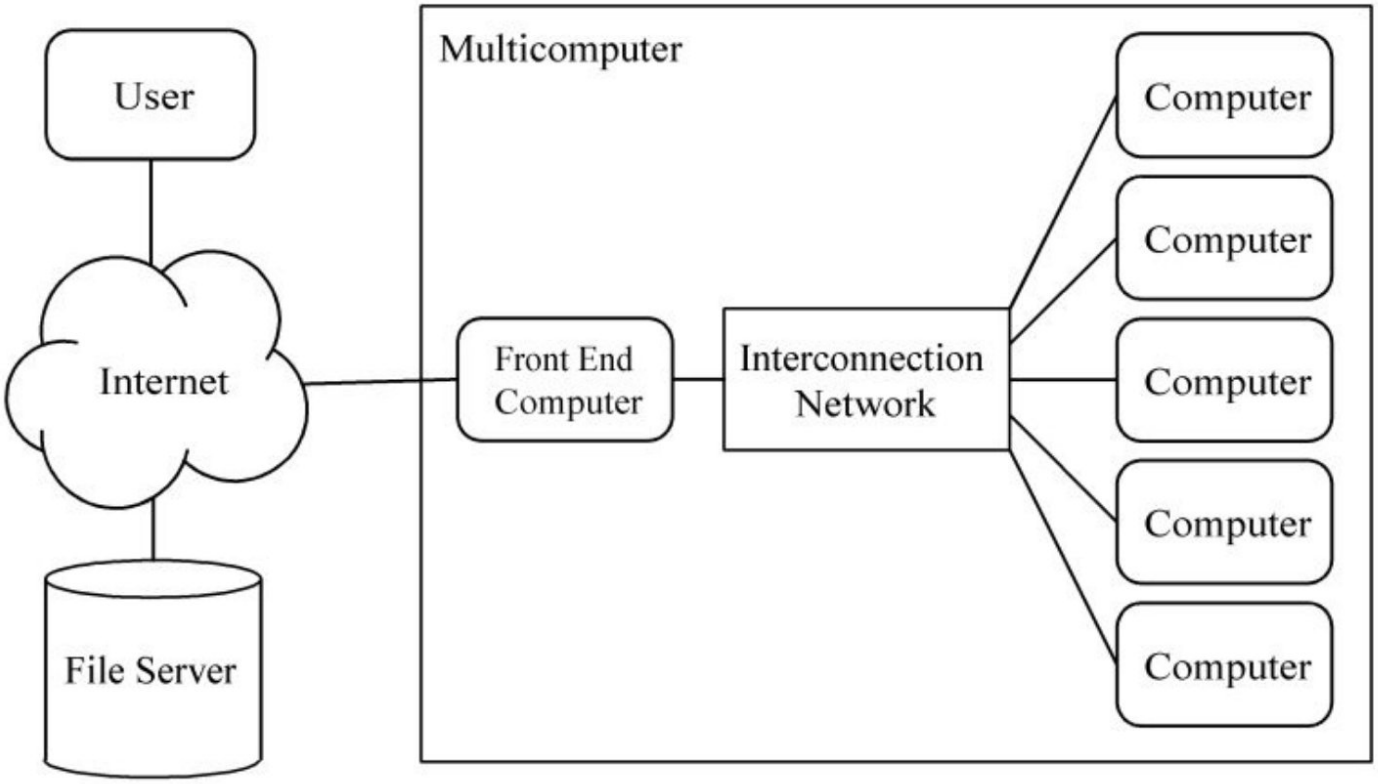
Processeur 0 libère X (flush)



3. Multi-ordinateurs

- Plusieurs ordinateurs ayant chacun sa propre mémoire
- Chaque ordinateur possède son propre espace d'adressage
- Les ordinateurs interagissent entre eux par l'envoie de messages
- Peut facilement être construit à partir d'ordinateurs conventionels et un réseau local.

Multi-ordinateurs asymétriques



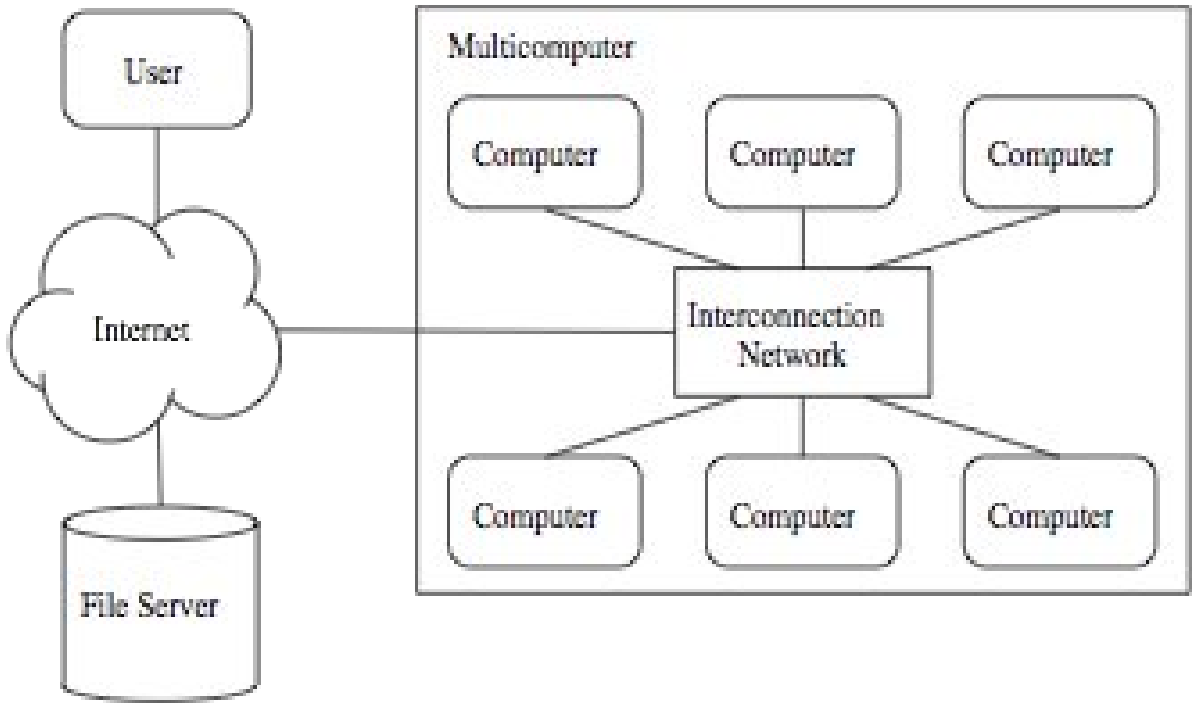
Avantages

- Les ordinateurs du réseaux sont dédiés au calcul parallèle et ne nécessite qu'un système d'exploitation simple
- Deux types de systèmes d'exploitations:
  - Ordinateur frontal
  - Ordinateurs du réseaux
- Système d'exploitation des ordinateurs du réseaux:
  - sans mémoire virtuel
  - sans entrée/sortie
  - facile à comprendre.

Désavantages

- Tout le système dépend de l'ordinateur frontal
- L'existence d'un unique ordinateur frontal limite le nombre d'ordinateurs
- Déboguage plus difficile
  - Les ordinateurs du réseau ne peuvent pas faire d'E/S
- Chaque application nécessite le développement de deux types de programmes (frontal et réseau)

Multi-ordinateurs symétriques



Avantages

- Élimine le problème du goulot d'étranglement causé par l'existence d'un ordinateur frontal unique
- Déboguage plus facile
- Chaque ordinateur exécute le même programme
  - Lorsqu'un seul ordinateur doit exécuter une opération, cela peut facilement être déterminé par une instruction conditionelle.

Désavantages

- Il est plus difficile de maintenir l'illusion de travailler sur une seule machine lorsqu'on peut se connecter sur n'importe quel ordinateur possédant son propre nom.
- Difficulté de balancer le travail entre les ordinateurs

Quel est le meilleur modèle?

- L’utilisation d’un système d’exploitation unique et complet facilitant le déboguage (e.g. Linux) plaide en faveur des multi-ordinateurs symétriques

- Les performances d’un processeurs dépendent beaucoup de l’efficacité de sa mémoire cache.
  - Cette situation est favorisé lorsqu’il n’y a qu’un seul processus par processeur
  - Cela plaide en faveur des multi-ordinateur asymétrique

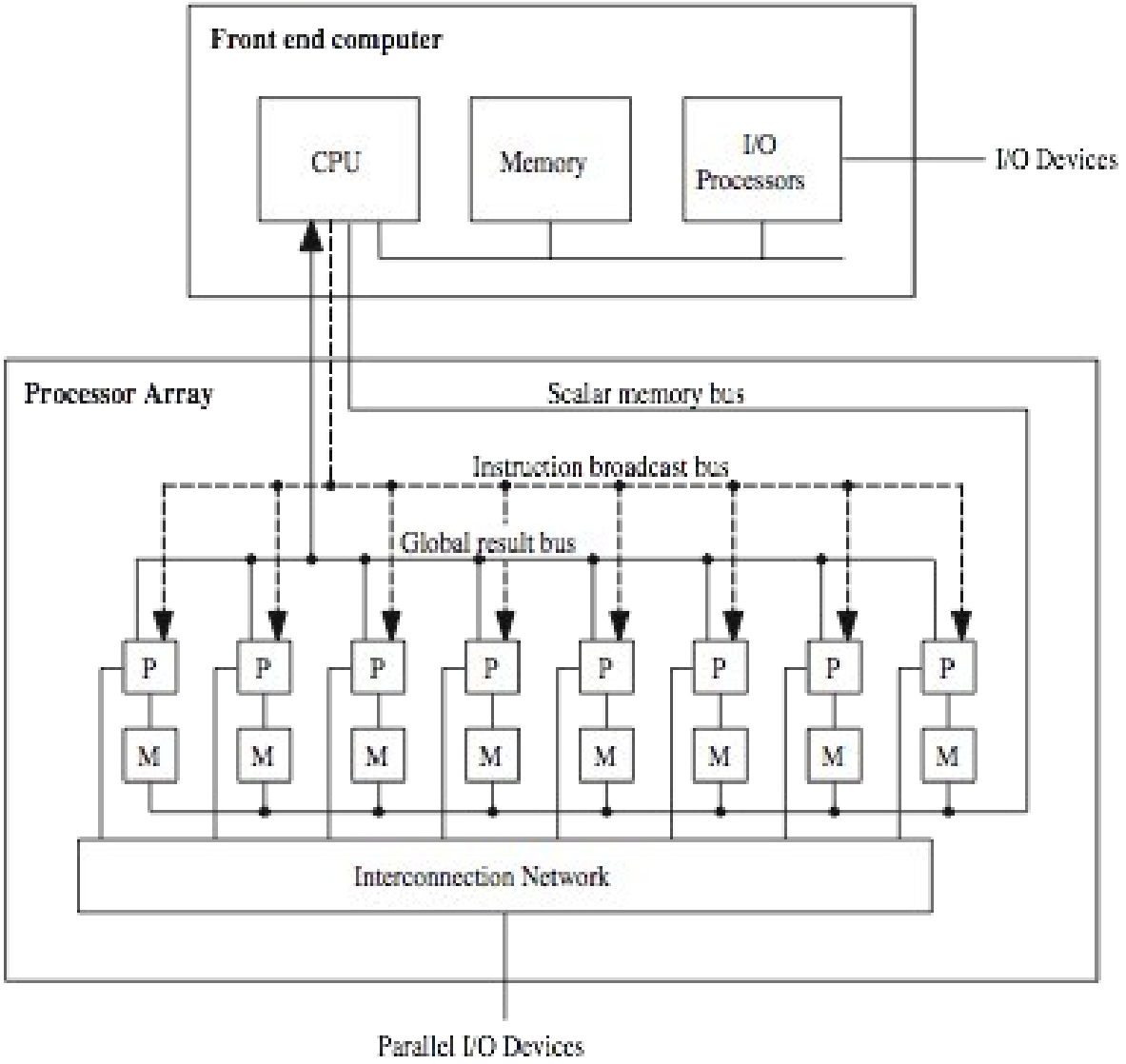
4. Ordinateurs vectoriels

- Inclut des opérations sur les vecteurs en plus des opérations sur les scalaires
- Deux types d'implémentations:
  - **Pipeline:** Les vecteur sont déplacés de la mémoire vers le processeurs en un flux de données traités en pipeline- CRAY-I, II, Cyber 205
  - **Réseau de processeurs:** Un ordinateur standard contient plusieurs processeurs identiques et synchronisés (ex. GPU)

Réseau de processeurs

- Ordinateur frontal
  - Programme
  - Données manipulées séquentiellement
- Réseau de processeurs
  - Mémoires locales
  - Données manipulées en parallèle
- L’ordinateur frontal diffuse les instructions aux processeurs du réseau
- Tous les processeurs du réseau exécutent la même instruction

Réseau de processeurs



Performance

- Quantité de travail par unité de temps
  - Ex. Nombre d'opérations par seconde
- La performance dépend de l'utilisation faite des processeurs.
- La performance est au maximum lorsque les processeurs sont utilisés à 100%

Exemple 1

- 1024 processeurs
- Chacun additionne un couple d'entiers en 1 sec
- On veut additionner deux vecteurs de1024-elements (un par processeur)

$$\text{Performance} = \frac{1024\ \text{operations}}{1\text{msec}} = 1,024 \times 10^9\ \text{ops/sec}$$

Exemple 2

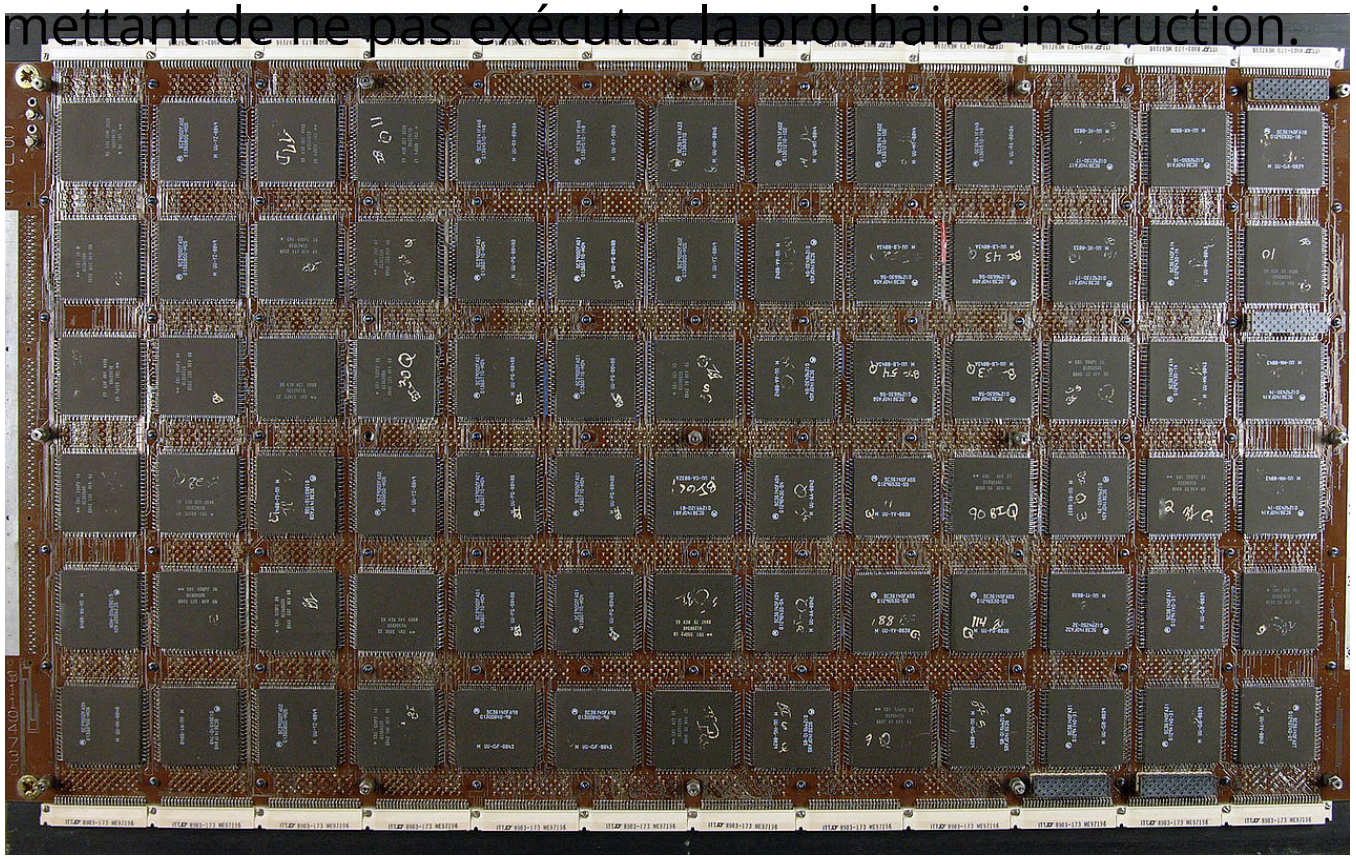
- 1024 processeurs
- Chacun peut additionner deux entiers en 1 sec
- On additionne deux vecteurs de 1025 éléments.

$$\text{Performance} = \frac{1025\,\text{operations}}{2\,\text{msec}} = 5,125 \times 10^{8}\,\text{ops/sec}$$

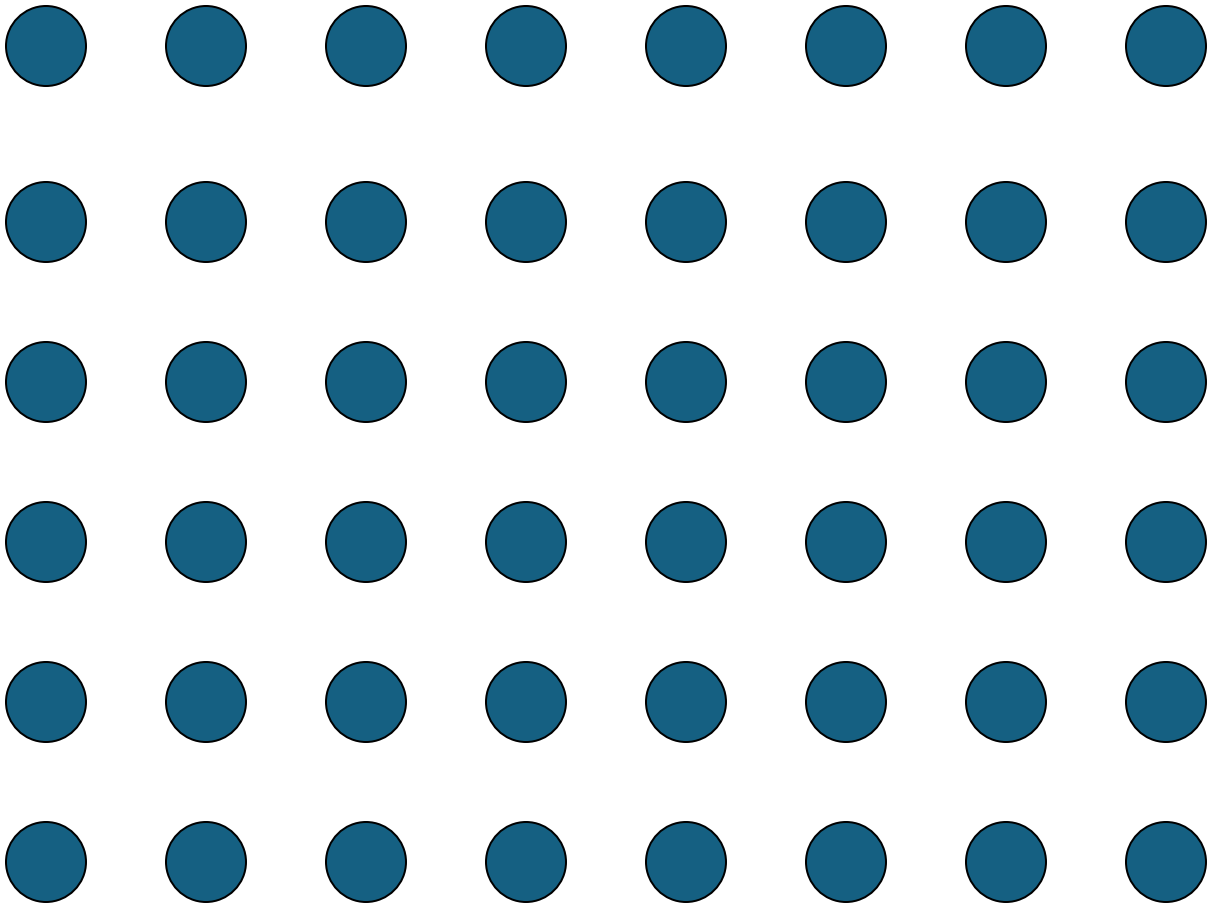
Activer et désactiver un processeur

Si tous les processeurs exécute la même instruction simultanément, comment traiter les instruction conditionnelles ???

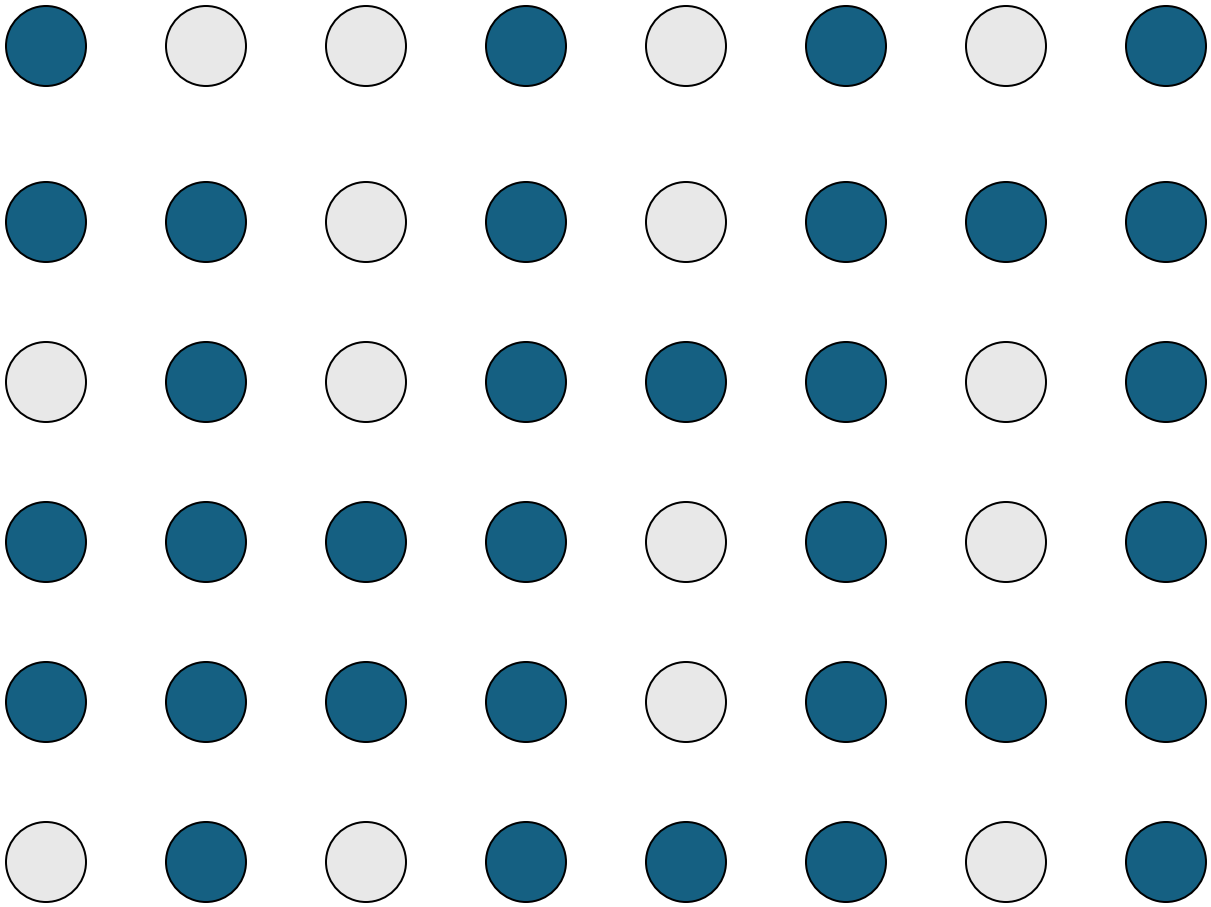
Chaque processeur possède un bit de masquage lui permettant de ne pas exécuter la prochaine instruction.



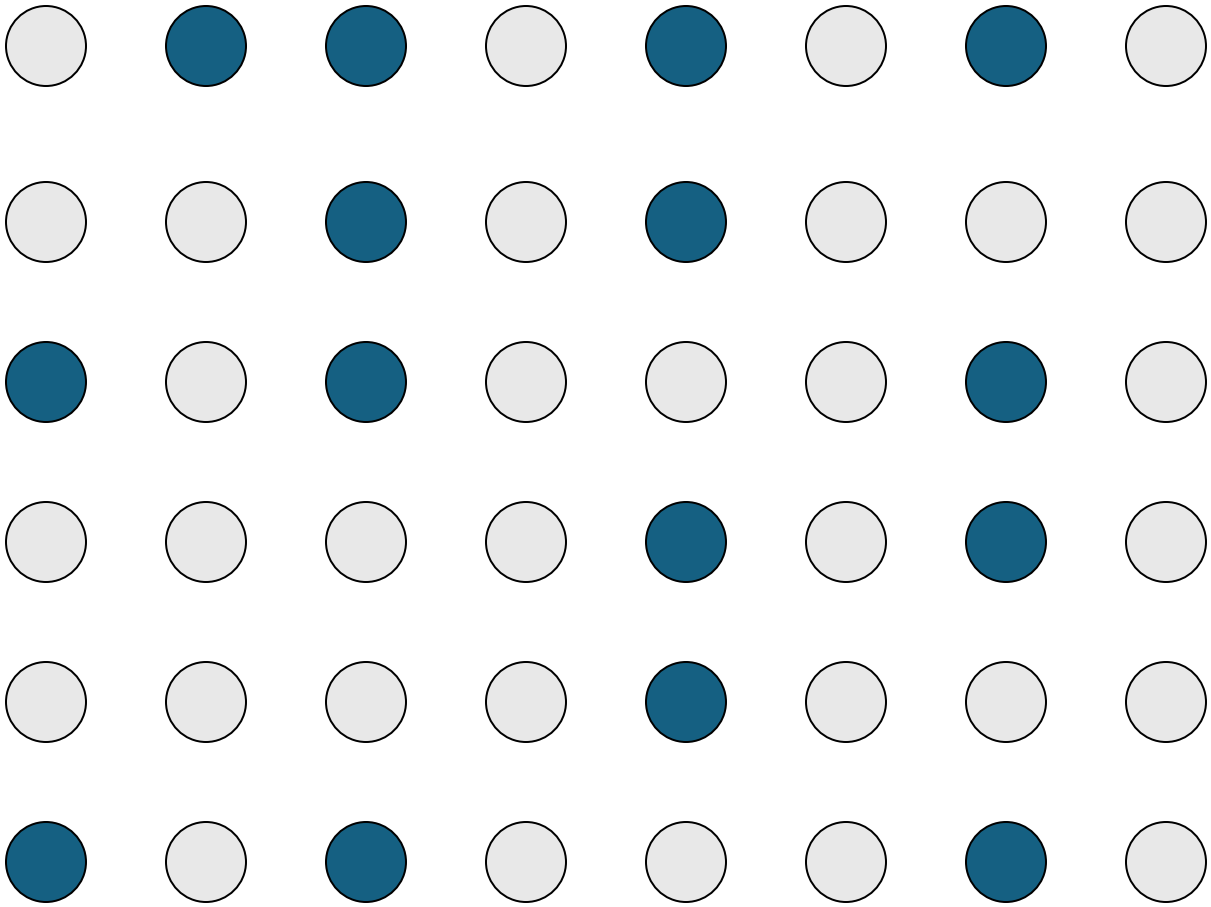
if (T[i]!=0) then T[i]=1 else T[i]=-1



if (T[i]!=0) then **T[i]=1** else T[i]=-1



if (T[i]!=0) then T[i]=1 else **T[i]=-1**



Désavantages des ordinateurs vectoriels.

- N’exploitent que le parallélisme de données
- La performance décroît en présence d’instructions conditionnelles
- S’adaptent mal à la présence de plusieurs usagers
- Nécessitent une bande passante très grande

5. Taxonomie de Flynn (1966)

- Flux d’instructions
- Flux de données
- Simple vs. multiple
- Quatre combinaisons
  - SISD
  - SIMD
  - MISD
  - MIMD

SISD

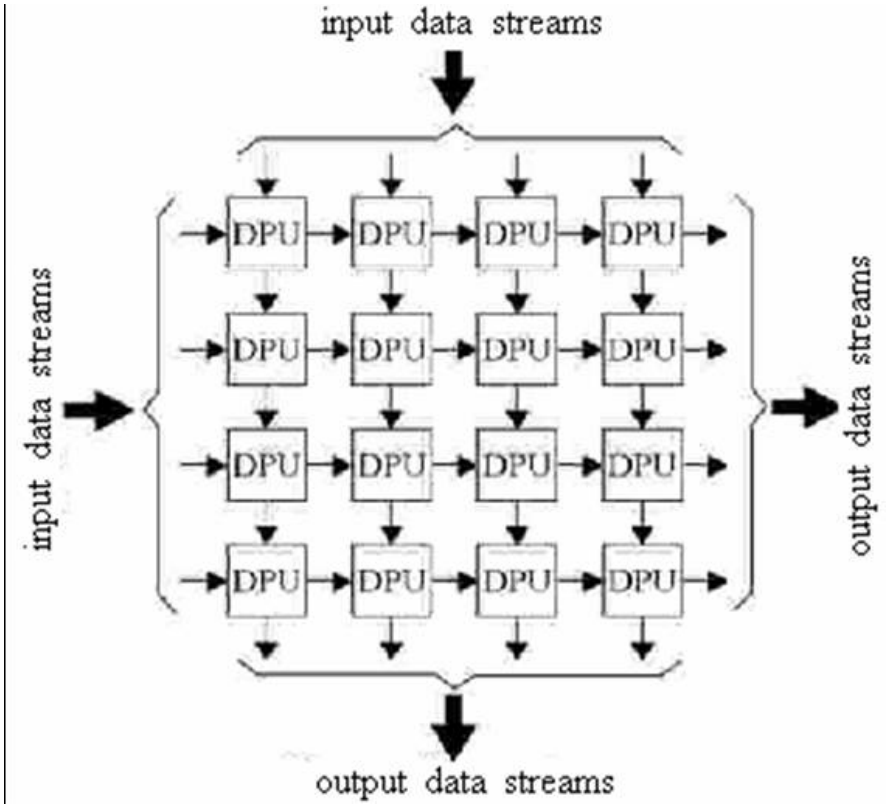
- Single Instruction, Single Data
- Système monoprocesseur
- Note: On ne compte pas les co-processeurs
  - Calcul en points flottant
  - Carte graphique
  - I/O
- Exemple: PC

SIMD

- Single Instruction, Multiple Data
- Ex. Processeurs vectoriels
  - GPU, Cray 1

MISD

- Multiple Instruction, Single Data
- Systèmes de tolérance aux fautes
- Peu d'exemples existent



MIMD

- Multiple Instruction, Multiple Data
  - Multiprocesseurs
  - Multi-ordinateurs