

# TP4 : Architecture des microprocesseurs

Júlia Ellen Dias Leite

julia-ellen.dias@ensta-paris.fr

Charbel Chahla

charbel.chahla@ensta.fr

Martin Brun

martin.brun@ensta.fr

Paul-henri DJOKO

paul-henri.djoko@ensta.fr

19 février 2026

## Analyse théorique de cohérence de cache

---

**Q1 :** En considérant que chaque thread s'exécute sur un processeur dans une architecture de type multicoeurs à base de bus et 1 niveau de cache (comme décrit Figure 21), décrivez le comportement de la hiérarchie mémoire et de la cohérence des caches pour l'algorithme de multiplication de matrices. On supposera que le thread principal se trouve sur le processeur d'indice 1.

## Paramètres de l'architecture multicoeurs

---

**Q2 :** Examinez le fichier de déclaration d'un élément de type « processeur superscalaire out- of-order », et présentez sous forme de tableau cinq paramètres configurables de ce type de processeur avec leur valeur par défaut. Choisissez de préférence des paramètres étudiés lors des séances TD/TP précédentes. Le fichier à consulter est le suivant :

```
$GEM5/src/cpu/o3/O3CPU.py
```

**Q3 :** Examinez le fichier d'options de la plateforme se.py, puis déterminez et présentez sous forme de tableau les valeurs par défaut des paramètres suivants :

- Cache de données de niveau 1 : associativité, taille du cache, taille de la ligne
- Cache d'instructions de niveau 1 : associativité, taille du cache, taille de la ligne
- Cache unifié de niveau 2 : associativité, taille du cache, taille de la ligne

Le fichier d'options à consulter est le suivant :

```
$GEM5/configs/common/Options.py
```

## Architecture multicoeurs avec des processeurs superscalaires in-order (Cortex A7)

---

**Q4 :** Déterminez quel est le processeur exécutant toujours le plus grand nombre de cycles. Expliquez pourquoi. Expliquez également pourquoi l'analyse du nombre de cycles sur ce processeur revient à analyser le nombre total de cycles d'exécution de l'application.

**Q5 :** Pour chaque configuration, quel est le nombre de cycles d'exécution de l'application ? Vous pourrez présenter vos résultats sous forme de graphe 2 axes

**Q6 :** Dédurre le speedup par rapport à la configuration à 1 thread.

**Q7 : En utilisant le nombre total d'instructions simulées, déterminez quelle est la valeur maximale de l'IPC pour chaque configuration ?**

**Q8 : Discussion et interprétation (max. 10 lignes).**

## Architecture multicoeurs avec des processeurs superscalaires out-of-order (Cortex A15)

**Q9 : Pour chaque configuration, quel est le nombre de cycles d'exécution de l'application ? Vous pourrez présenter vos résultats sous forme de graphe 3 axes.**

Temps d'exécution simulé (sim\_seconds) en fonction du nombre de threads et de la largeur du processeur :

TABLE 1 – Temps d'exécution (sim\_seconds) par largeur de processeur et nombre de threads.

Width	1 thread	2 threads	4 threads	8 threads	16 threads
2 voies	0.000169	0.000115	0.000089	0.000075	0.000071
4 voies	0.000116	0.000087	N/A	0.000066	0.000065
8 voies	0.000114	0.000086	0.000072	0.000066	N/A

Les résultats montrent que :

- Le temps d'exécution diminue avec l'augmentation du nombre de threads (parallelism).
- Pour un nombre de threads donné, les configurations 4 et 8 voies sont en général plus rapides que 2 voies.
- Le meilleur temps observé est **0.000065 s** (4 voies, 16 threads), contre **0.000169 s** (2 voies, 1 thread).
- Certaines combinaisons sont absentes (N/A) : 4 voies / 4 threads et 8 voies / 16 threads.

*Remarque :* Le fichier de résultats inclut aussi `sim_ticks`, `cycles_max` et `insts_max`, ce qui permet d'analyser les tendances en cycles en plus du temps simulé.

**Q10 : Dédurre le speedup par rapport à la configuration à 1 thread.**

Le **speedup** est défini comme le rapport du temps de base (1 thread) au temps pour  $n$  threads :

$$\text{Speedup}(n) = \frac{T_1}{T_n}$$

TABLE 2 – Speedup par rapport à 1 thread.

Width	2	4	8	16
2 voies	1.47	1.90	2.25	2.38
4 voies	1.33	N/A	1.76	1.78
8 voies	1.33	1.58	1.73	N/A

- Le speedup est **sub-linéaire** : il ne croît pas proportionnellement au nombre de threads.
- En 2 voies, le speedup progresse jusqu'à **2.38×** à 16 threads.
- En 4 voies, on observe **1.78×** à 16 threads, avec une donnée manquante à 4 threads.
- En 8 voies, le speedup atteint **1.73×** à 8 threads (valeur 16 threads indisponible).
- Les écarts entre largeurs et la saturation du gain confirment une contention sur les ressources partagées (cache/mémoire/synchronisation).

**Q11 : En utilisant le nombre total d'instructions simulées, déterminez quelle est la valeur maximale de l'IPC pour chaque configuration ?**

L'**IPC (Instructions Per Cycle)** mesure le parallélisme au niveau des instructions exécutées par cycle d'horloge.

TABLE 3 – IPC par largeur du processeur et nombre de threads.

Width	1 thread	2 threads	4 threads	8 threads	16 threads
2 voies	1.351054	1.800073	1.751922	1.686834	1.578735
4 voies	1.963032	3.199570	N/A	2.785343	2.659370
8 voies	2.000776	3.379401	3.345791	3.286575	N/A

- L'IPC varie avec la largeur : à 1 thread, il passe de **1.351** (2 voies) à **2.001** (8 voies).
- En multi-thread, les IPC maximales les plus élevées sont obtenues en 8 voies (jusqu'à **3.380** à 2 threads).
- Les résultats 4 voies et 8 voies sont proches à partir de 8 threads (~2.79 vs ~3.29), ce qui suggère une saturation progressive.
- Deux combinaisons restent indisponibles dans les mesures (4 voies/4 threads et 8 voies/16 threads).

### Q12 : Discussion et interprétation (max. 10 lignes)

L'étude des performances du DerivO3CPU avec différents widths et nombres de threads révèle plusieurs points clés :

**2) Parallelism limité par la contention :** Le speedup reste sub-linéaire pour toutes les largeurs, ce qui indique des limites côté mémoire/cache/synchronisation malgré l'augmentation du nombre de threads.

**3) Données partielles :** Deux points expérimentaux manquent (4 voies/4 threads et 8 voies/16 threads). Les tendances globales restent néanmoins cohérentes : gains nets entre 2 et 4/8 voies, puis saturation progressive aux forts niveaux de parallélisme.

## Configuration CMP la plus efficace

**Q13 : Proposez une configuration ou une gamme de configuration de l'architecture CMP (nombre de threads de l'application test omp, nombre et type de coeurs) qui vous semble la plus appropriée si la contrainte recherchée par le concepteur du système est l'efficacité surfacique ? Discussion et interprétation (max. 10 lignes).**

**Q14 : Au regard de l'évolution théorique du speedup et son évolution constatée lors des questions précédentes, proposez une tentative d'explication (max. 10 lignes).**