

Multiple-Choice Questions - Lecture 5

Question 1:

Which memory type in CUDA has the lowest latency? / Quel type de mémoire dans CUDA a la latence la plus faible?

- a. Global Memory / a. Mémoire globale
- b. Shared Memory / b. Mémoire partagée
- c. Constant Memory / c. Mémoire constante
- d. Texture Memory / d. Mémoire de texture

Réponse / Answer : b)

Question 2:

What is the main benefit of shared memory in CUDA? / Quel est le principal avantage de la mémoire partagée dans CUDA?

- a. Faster global memory writes / a. Écritures plus rapides en mémoire globale
- b. Reduction in bank conflicts / b. Réduction des conflits de banque
- c. Low latency and high bandwidth / c. Faible latence et large bande passante
- d. Increases global memory size / d. Augmente la taille de la mémoire globale

Réponse / Answer : c)

Question 3:

What is a bank conflict in CUDA shared memory? / Qu'est-ce qu'un conflit de banque dans la mémoire partagée CUDA?

- a. Multiple threads accessing the same bank address / a. Plusieurs threads accédant à la même adresse de banque
- b. All threads accessing the same memory location / b. Tous les threads accédant à la même adresse mémoire
- c. Excessive usage of shared memory / c. Utilisation excessive de la mémoire partagée
- d. Inefficient memory allocation / d. Allocation inefficace de mémoire

Réponse / Answer : a)

Question 4:

How can bank conflicts be avoided in shared memory? / Comment éviter les conflits de banque dans la mémoire partagée?

- a. Use padding between memory elements / a. Utiliser du rembourrage entre les éléments de mémoire
- b. Allocate more global memory / b. Allouer plus de mémoire globale
- c. Synchronize threads before accessing memory / c. Synchroniser les threads avant d'accéder à la mémoire
- d. Use read-only cache instead / d. Utiliser à la place un cache en lecture seule

Réponse / Answer : a)

Question 5:

What does __syncthreads() do in CUDA? / Que fait __syncthreads() dans CUDA?

- a. Terminates a kernel / a. Termine un kernel
- b. Synchronizes threads within a block / b. Synchronise les threads au sein d'un bloc
- c. Allocates shared memory dynamically / c. Alloue de la mémoire partagée de manière dynamique
- d. Optimizes global memory access / d. Optimise l'accès à la mémoire globale

Réponse / Answer : b)

Question 6:

What is the purpose of padding in shared memory? / Quel est l'objectif du rembourrage dans la mémoire partagée?

- a. To reduce memory usage / a. Réduire l'utilisation de la mémoire
- b. To avoid bank conflicts / b. Éviter les conflits de banque
- c. To increase latency / c. Augmenter la latence
- d. To allocate memory dynamically / d. Allouer la mémoire dynamiquement

Réponse / Answer : b)

Question 7:

What type of access pattern minimizes bank conflicts in shared memory? / Quel type de modèle d'accès minimise les conflits de banque dans la mémoire partagée?

- a. Column-major / a. Colonne principale

- b. Random / b. Aléatoire
- c. Row-major / c. Ligne principale
- d. Mixed / d. Mixte

Réponse / Answer : c)

Question 8:

What does the `__shared__` qualifier in CUDA indicate? / Que signifie le qualificateur `__shared__` dans CUDA?

- a. Global memory / a. Mémoire globale
- b. Shared memory / b. Mémoire partagée
- c. Constant memory / c. Mémoire constante
- d. Read-only memory / d. Mémoire en lecture seule

Réponse / Answer : b)

Question 9:

What is the effect of `__threadfence()` in CUDA? / Quel est l'effet de `__threadfence()` dans CUDA?

- a. Ensures thread synchronization within a warp / a. Assure la synchronisation des threads dans un warp
- b. Ensures memory consistency across the grid / b. Assure la cohérence mémoire dans la grille
- c. Reduces latency in memory accesses / c. Réduit la latence dans les accès mémoire
- d. Enables dynamic memory allocation / d. Permet l'allocation de mémoire dynamique

Réponse / Answer : b)

Question 10:

How is constant memory declared in CUDA? / Comment la mémoire constante est-elle déclarée dans CUDA?

- a. Using `__global__` / a. En utilisant `__global__`
- b. Using `__shared__` / b. En utilisant `__shared__`
- c. Using `__constant__` / c. En utilisant `__constant__`
- d. Using `__device__` / d. En utilisant `__device__`

Réponse / Answer : c)