M2 - Architecture et Programmation d'accélérateurs Matériels.

(APM 2022-2023)

TD1

CUDA

julien.jaeger@cea.fr
adrien.roussel@cea.fr

Les objectifs de ce TD sont :
— Compréhension du modèle déxécution
— Emulation sur CPU
— Calcul d'indice global

I Calcul d'indice global

Q.1: En utilisant les variables prédéfinies par CUDA dans un kernel, calculer l'indice global d'un block dans une grille 2D

Correction

bloc1 = blockIdx.y * gridDim.x + blockIdx.x.

Q.2: En utilisant les variables prédéfinies par CUDA dans un kernel, calculer l'indice global d'un block dans une grille 3D

Correction

bloc2 = blockIdx.z * gridDim.y * gridDim.x + blockIdx.y * gridDim.x + blockIdx.x.

Q.3: En utilisant les variables prédéfinies par CUDA dans un kernel, calculer la taille (nombre de threads) d'un bloc défini en 3D

Correction

size = blockDim.x * blockDim.y * blockDim.z

Q.4: En utilisant les variables prédéfinies par CUDA dans un kernel, calculer l'indice global d'un thread dans un bloc 3D

Correction

th = threadIdx.z * blockDim.y * blockDim.x + threadIdx.y * blockDim.x + threadIdx.x.

Q.5: En vous servant des questions précédentes, calculer l'indice global d'un thread dans une grille 3D composée de blocs 3D

Correction

indice_global = bloc2*size+th

II Modèle d'exécution et SDK CUDA

Dans cette partie nous considérons le fichier td2.cu.

Q.6: Quelle partie du programme doit s'exécuter sur l'hôte ? Quelle partie sur le device ?

Correction

La fonction précédée par le mot clé *___global___* définit un kernel. Par conséquent, cette fonction va s'exécuter sur le device (i.e. la carte GPGPU). Le reste du programme va s'exécuter sur l'hôte (i.e. sur le CPU). Les fonctions prefixées cuda* sont exécutées par l'hôte mais permettent de piloter le device.

Q.7: Que calcule ce programme ? (si vous ne savez pas répondre à cette question, répondre à la suivante pourra vous aider).

Correction

Ce programme calcule l'addition de deux vecteurs h_a et h_b de taille N et affecte le résultat dans h_c (voir fichier *CORR/tp1_check.cu*).

Q.8: Combien y a-t-il de blocs au total ? Combien de threads par blocs ? Combien de threads au total ?

Correction

Chaque bloc contient 64 threads (*=dimBlock.x*dimBlock.y*dimBlock.z =64*1*1*). Le nombre de blocs est égal à 16 (*=(1000+63)/63*1*1 =dimGrid.x*dimGrid.y*dimGrid.z*) ce qui représente 1024 threads en tout (=16*64 =dimBlock.x*dimGrid.x). 16 blocs est le plus petit nombre de blocs de 64 threads qui contient au moins 1000 (=N) threads.

Q.9: Émuler sur CPU le comportement du GPU sans utiliser le SDK CUDA. Pour ce faire, réécrire le programme en C/C++ avec les contraintes suivantes :

1. utilisation de nouveaux tableaux à utiliser pour le "kernel"
2. copie des données en entrées et en sorties du "kernel"
3. utilisation d'une fonction `kernel`
4. utillisation des grilles de blocs et de threads
5. calcul d'un indice global pour accéder aux cases des tableaux

Correction

Pour l'émulation sur CPU, voir fichier *CORRECTIONS/Partie2/td2.c*.