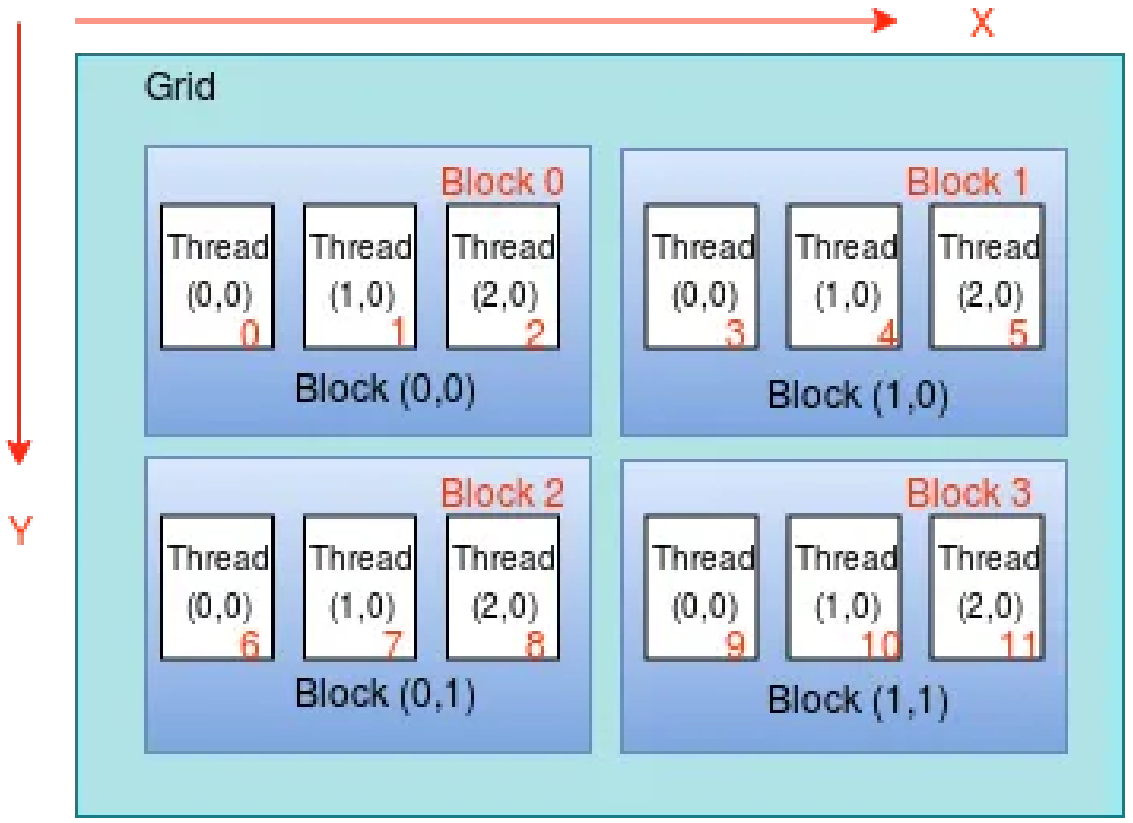
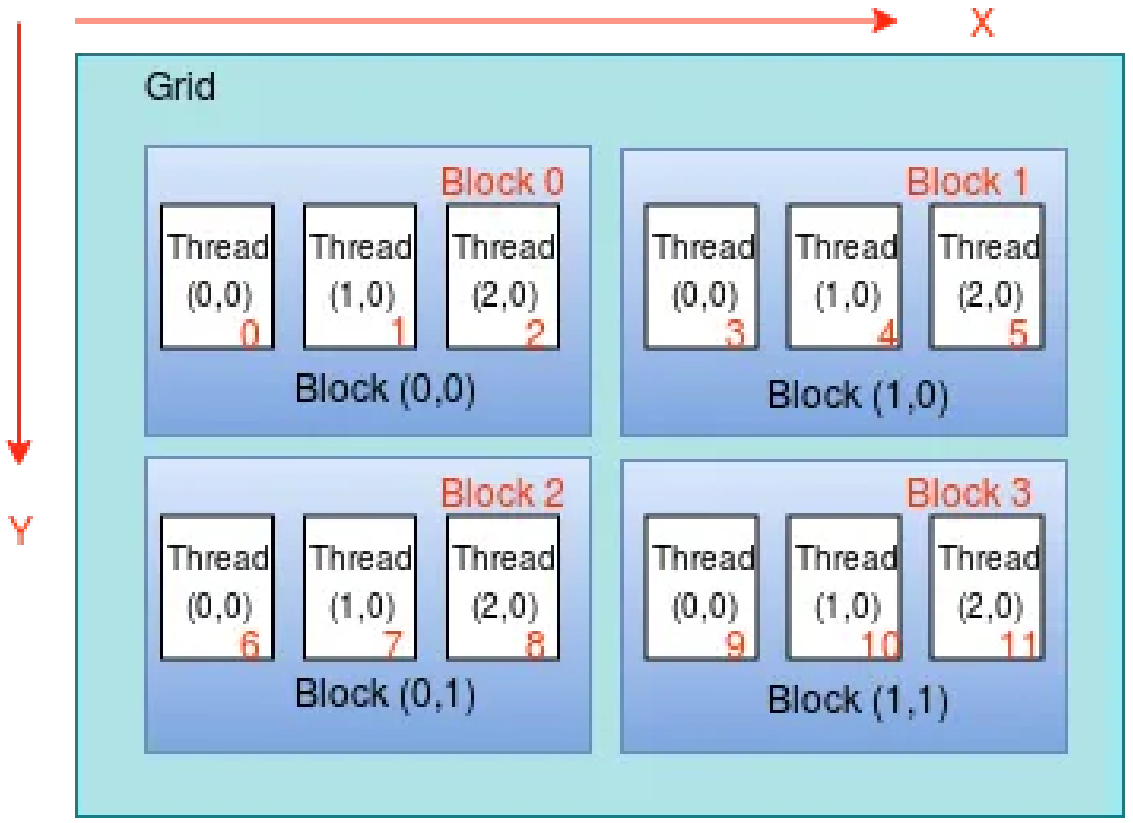
TD1

Modèle d’exécution CUDA

Calcul d’indice global: grille 2D



Calcul d’indice global: grille 2D

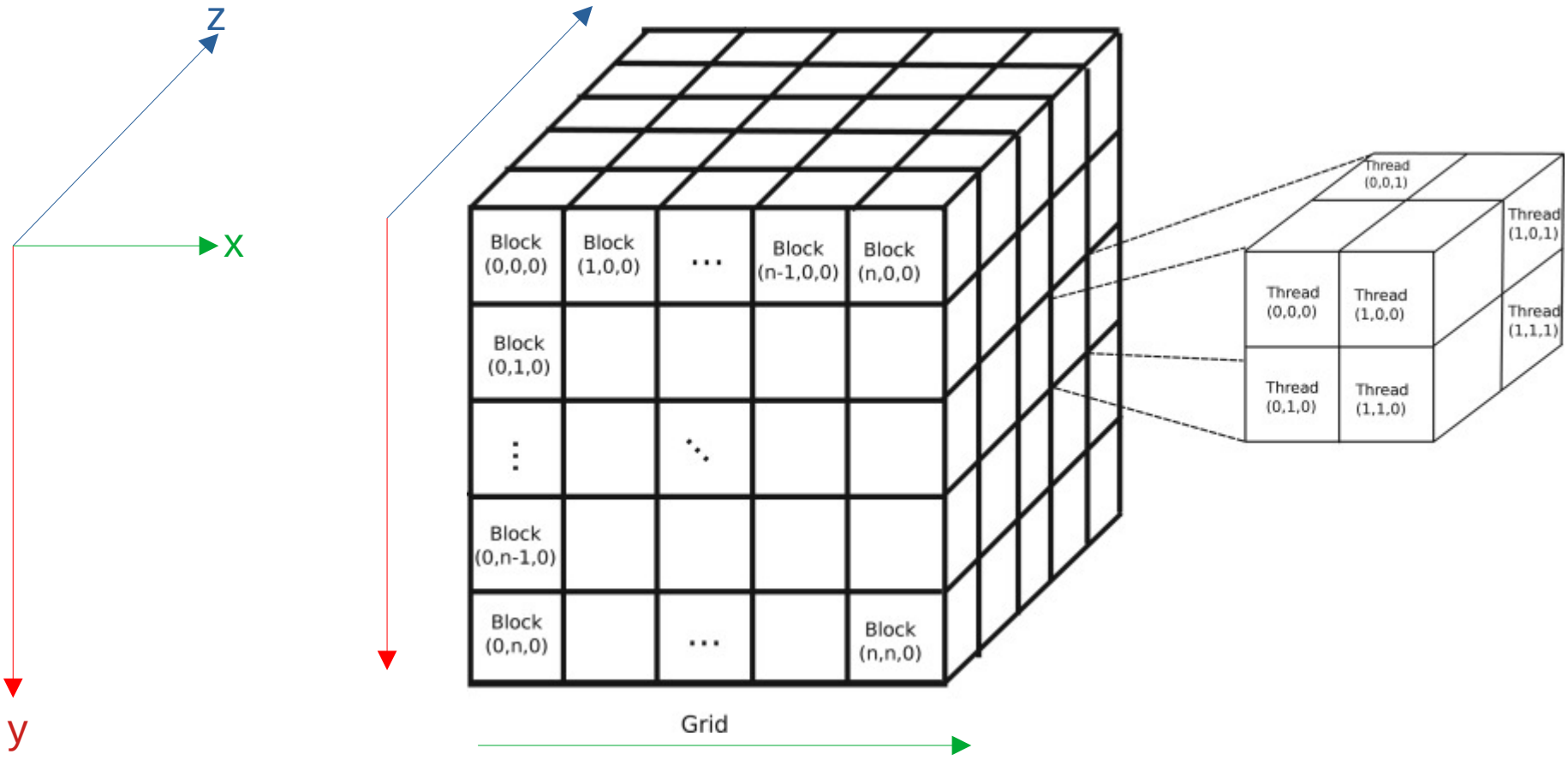


Q1)

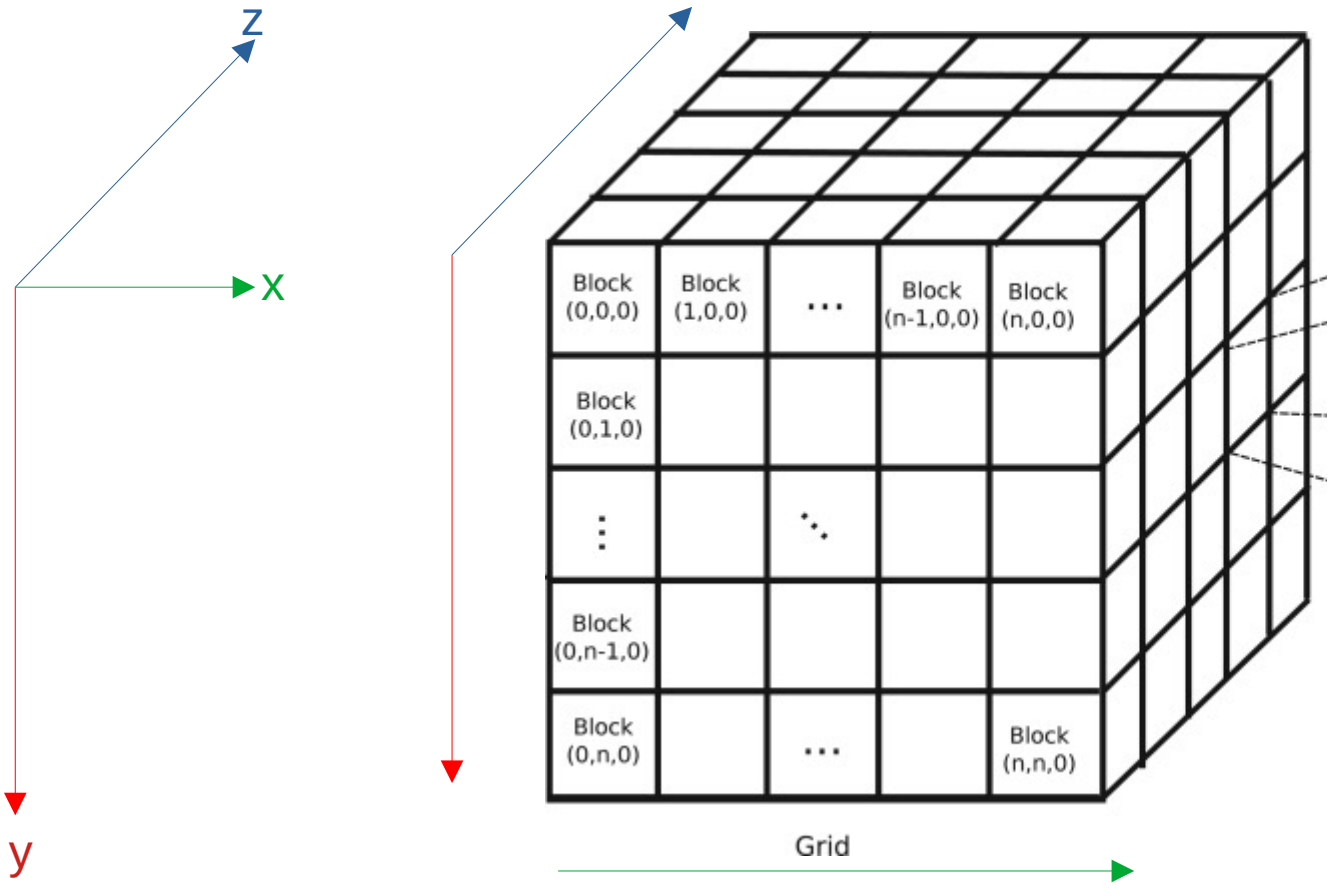
block_id =
blockIdx.y * gridDim.x
+ blockIdx.x

décalage à la bonne ligne
+ id local dans la ligne

Calcul d’indice global: grille 3D

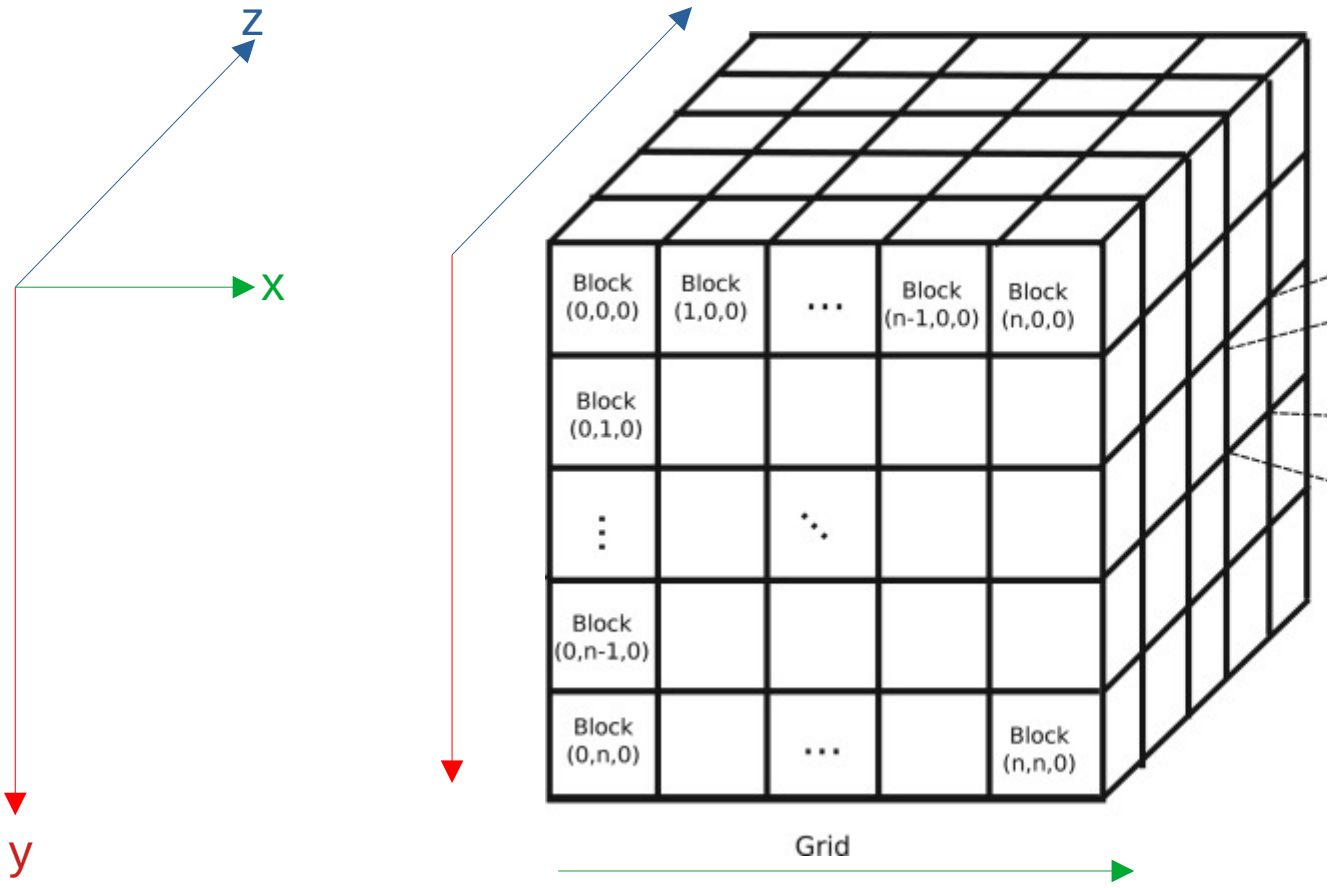


Calcul d'indice global: grille 3D



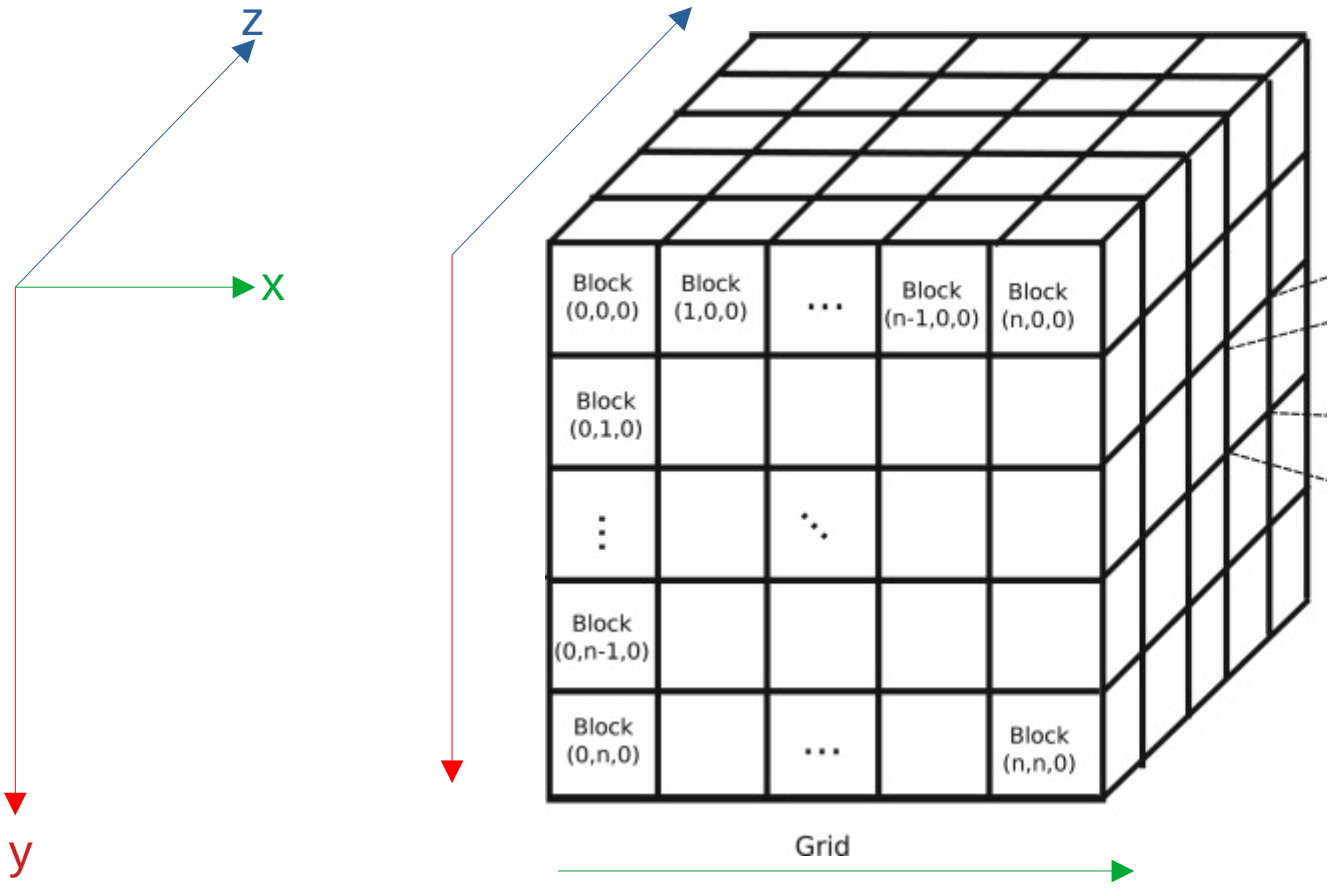
block_id =
décalage dans la bonne tranche
+ ...
+ ...

Calcul d’indice global: grille 3D



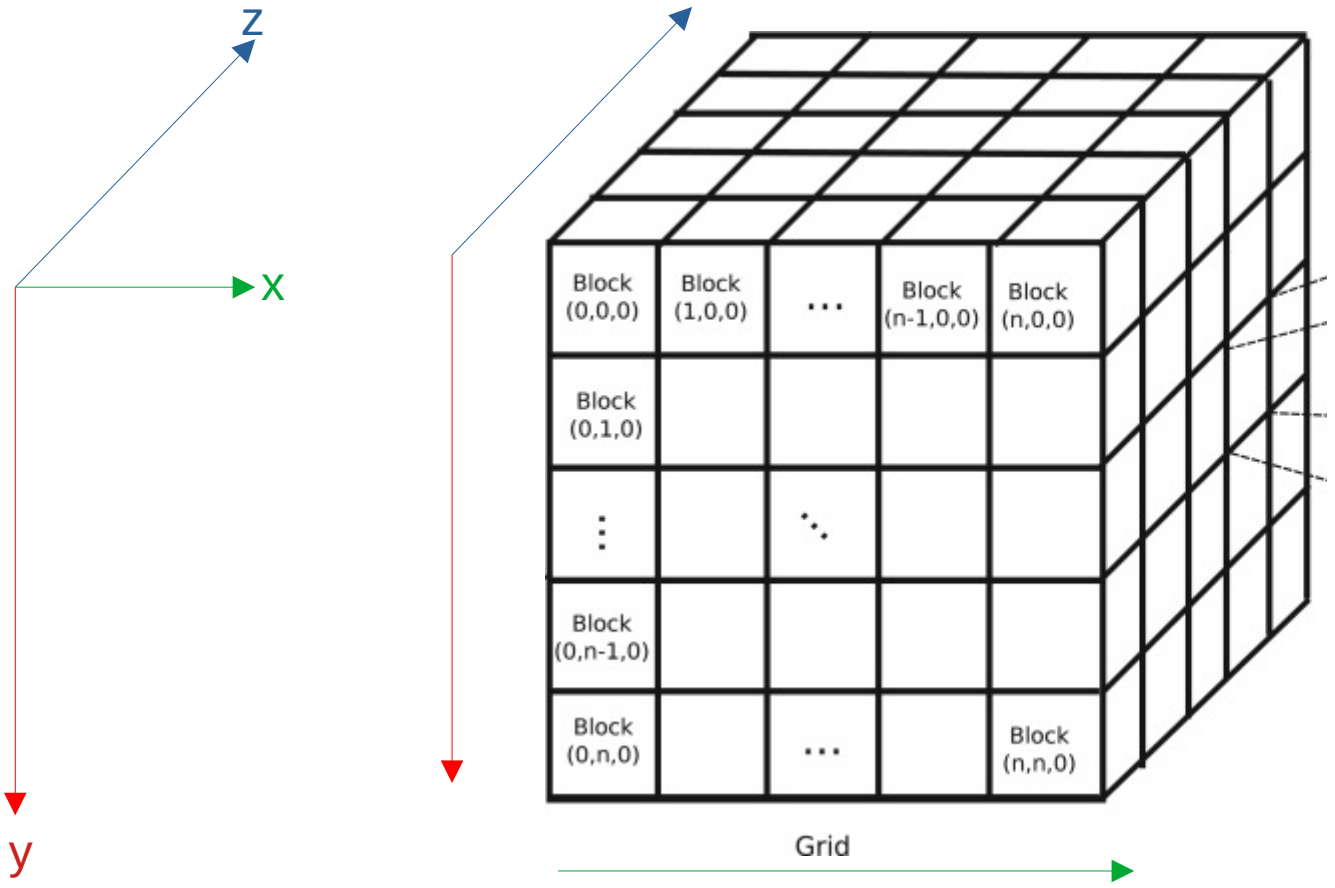
block_id =
décalage dans la tranche z = blockIdx.z
+ ...

Calcul d’indice global: grille 3D



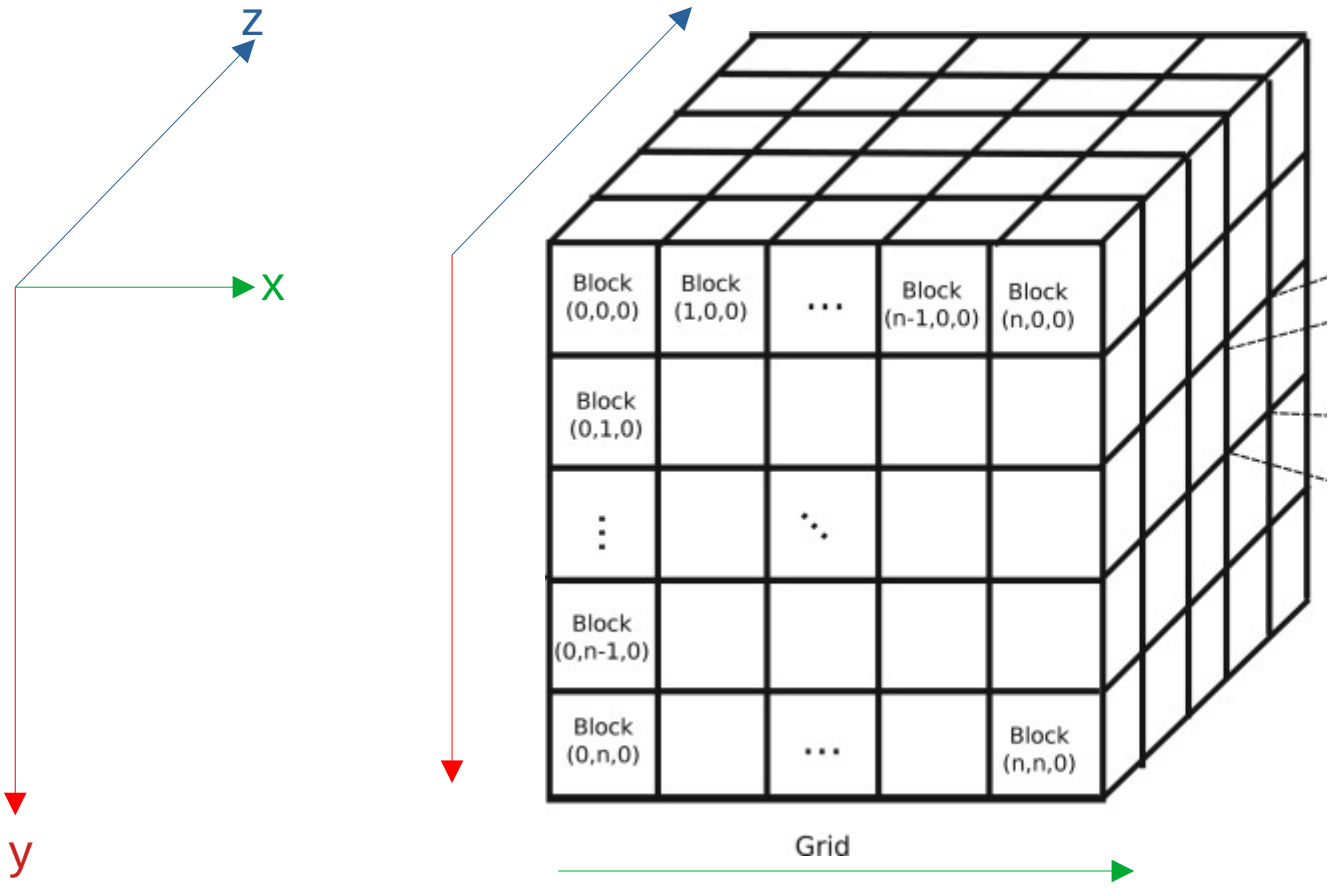
block_id =
blockIdx.z *
dim(tranche z = cte)
+ ...

Calcul d’indice global: grille 3D



block_id =
(blockIdx.z * gridDim.x
* gridDim.y)
+ ...

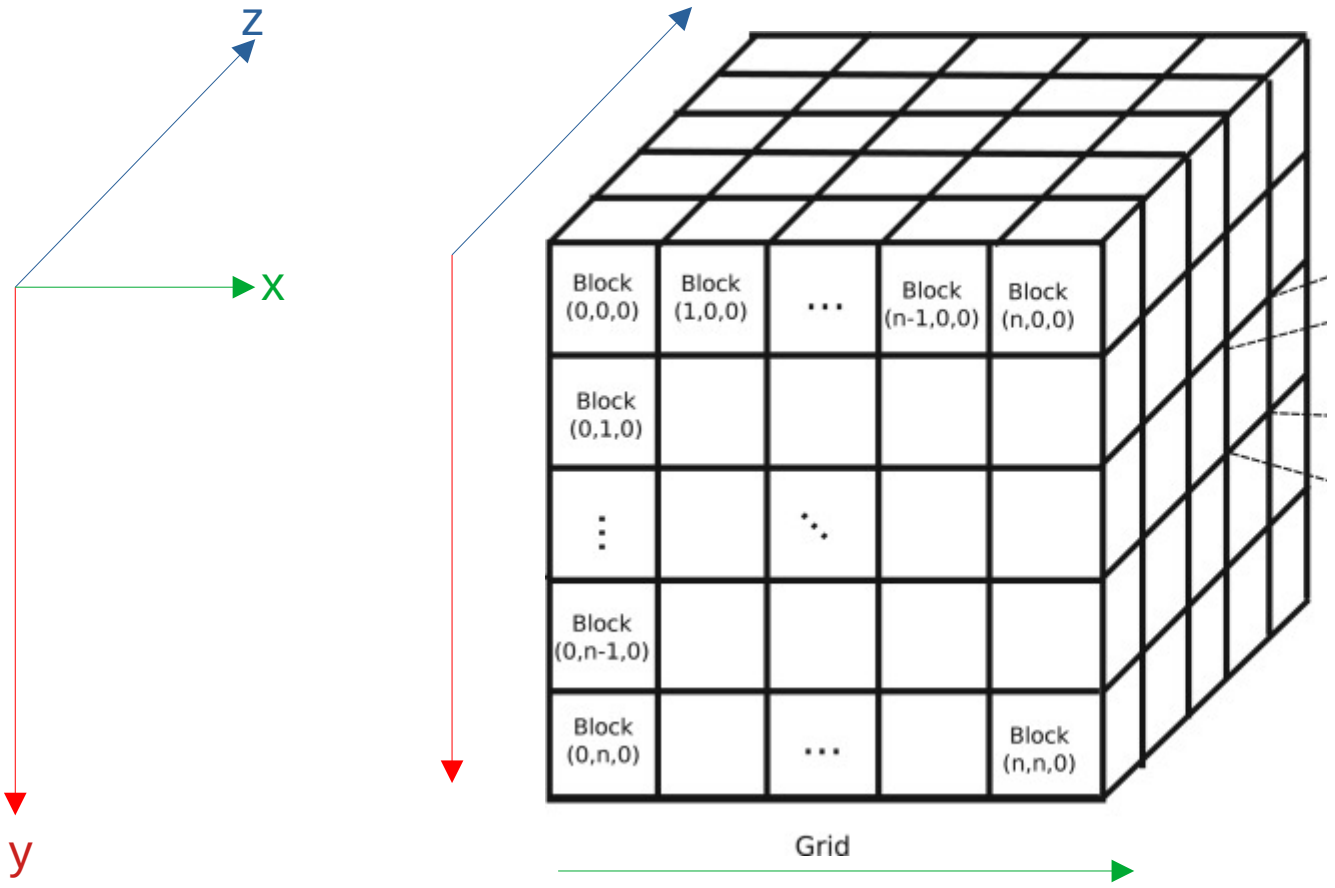
Calcul d’indice global: grille 3D



block_id = (blockIdx.z * gridDim.x * gridDim.y) + indice dans la tranche

Cf Q1

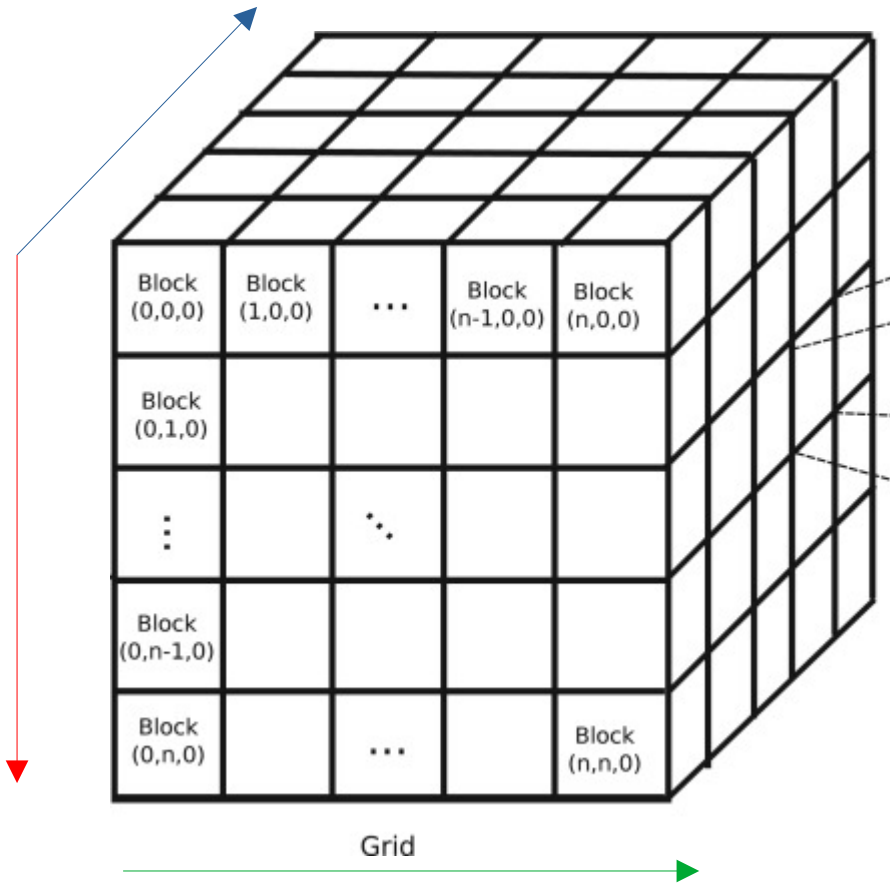
Calcul d'indice global: grille 3D



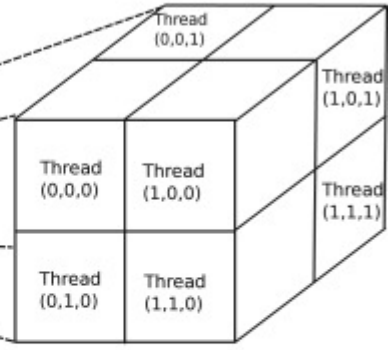

```
block_id =
(blockIdx.z * gridDim.x
*  gridDim.y)
+ blockIdx.y *
gridDim.x
+ blockIdx.x
```

Q2

Calcul d’indice global: grille 3D

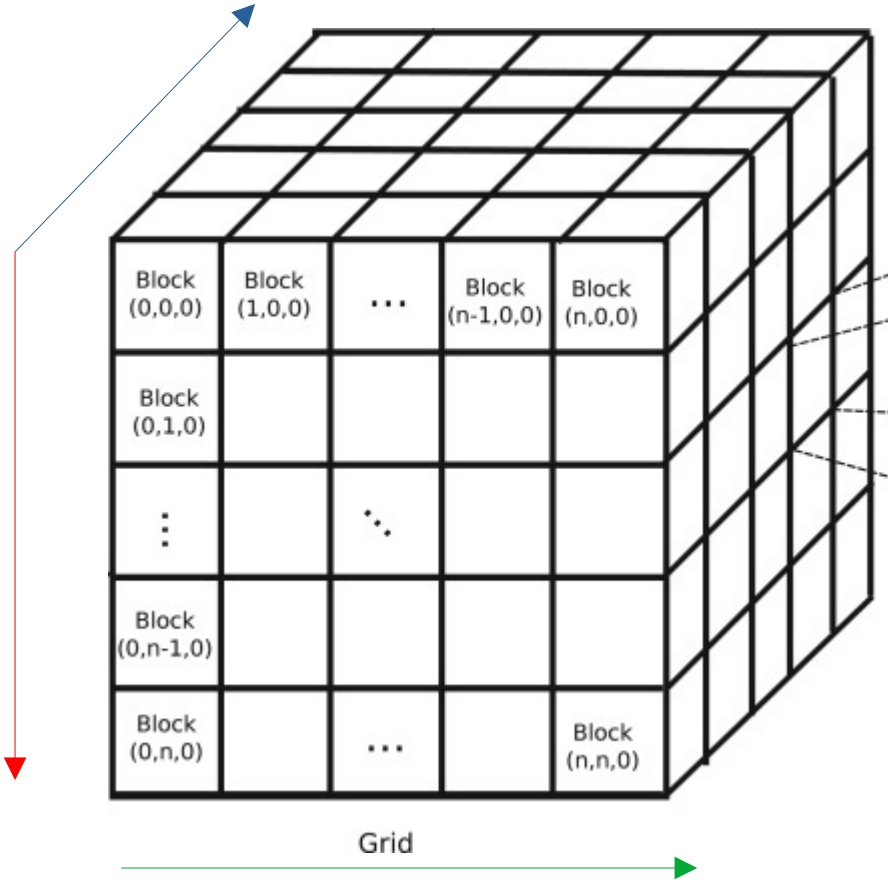


Q3

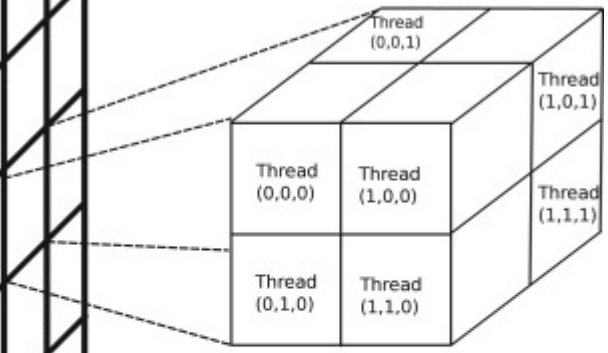


Taille d’un bloc
size = (blockDim.x * blockDim.y * blockDim.z)

Calcul d’indice global: grille 3D

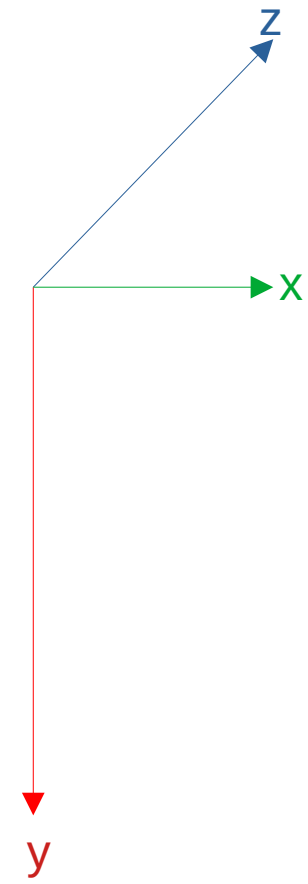


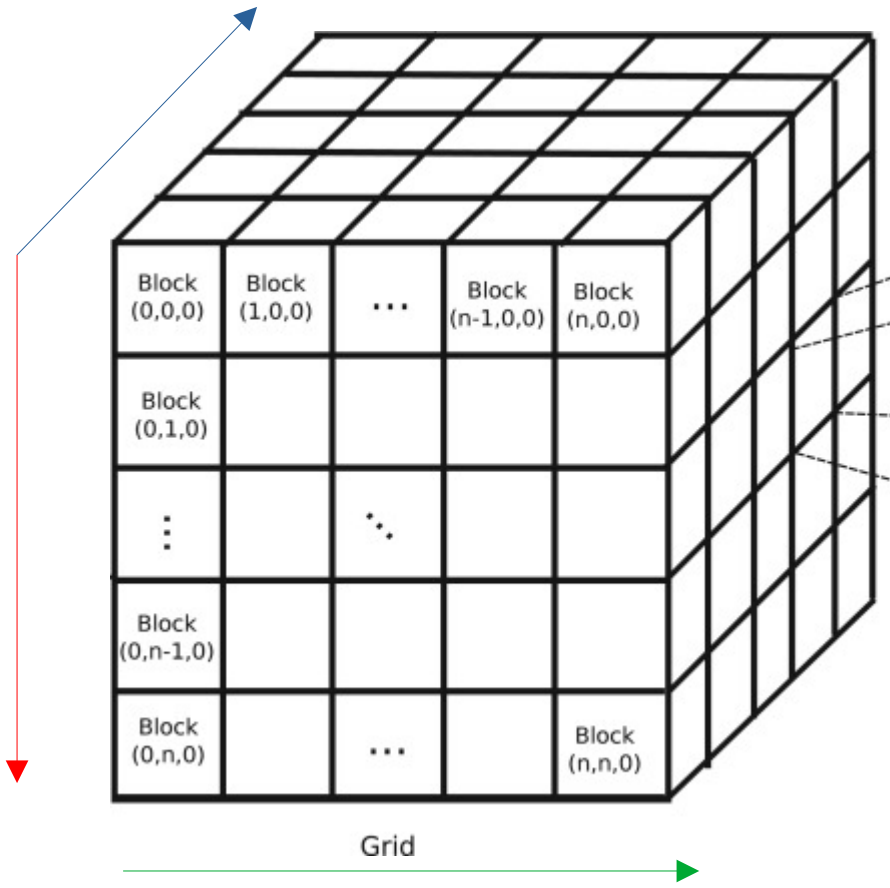
Q4



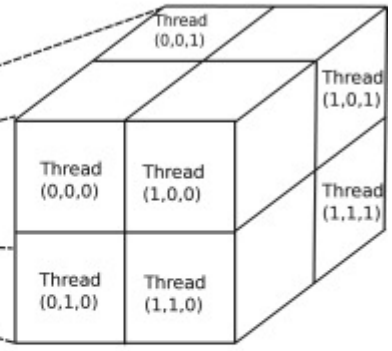
Indice d’un thread dans un bloc: idem Q2 avec
*** gridDim → blockDim**
*** blockIdx → threadIdx**

Calcul d’indice global: grille 3D



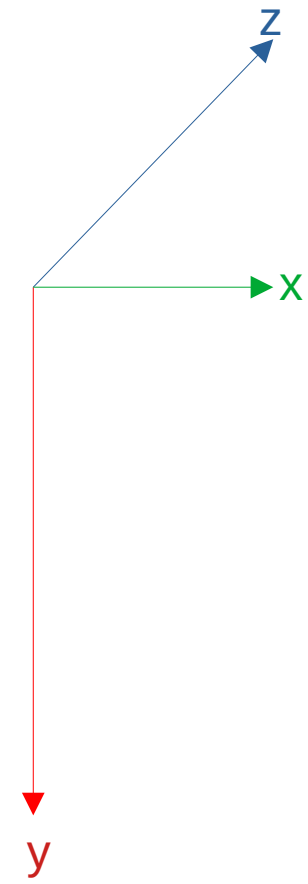


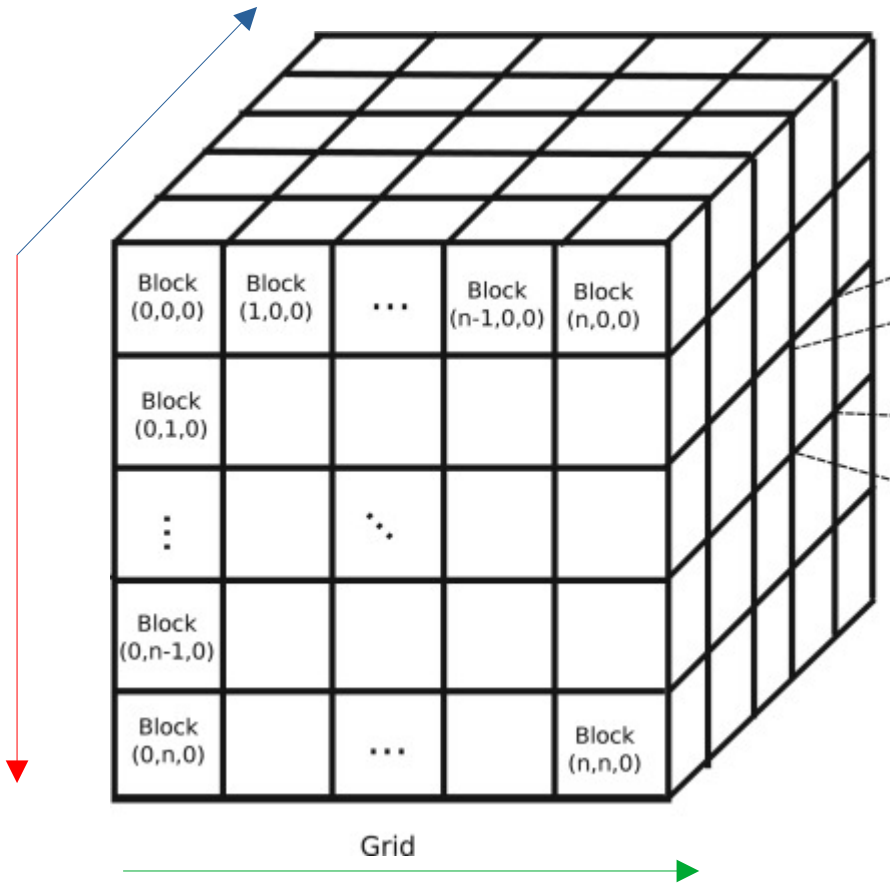
Q4



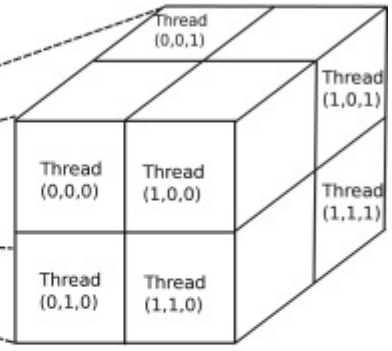

```
th = (threadIdx.z *
blockDim.y * blockDim.x) +
threadIdx.y * blockDim.x
+ threadIdx.x
```

Calcul d’indice global: grille 3D



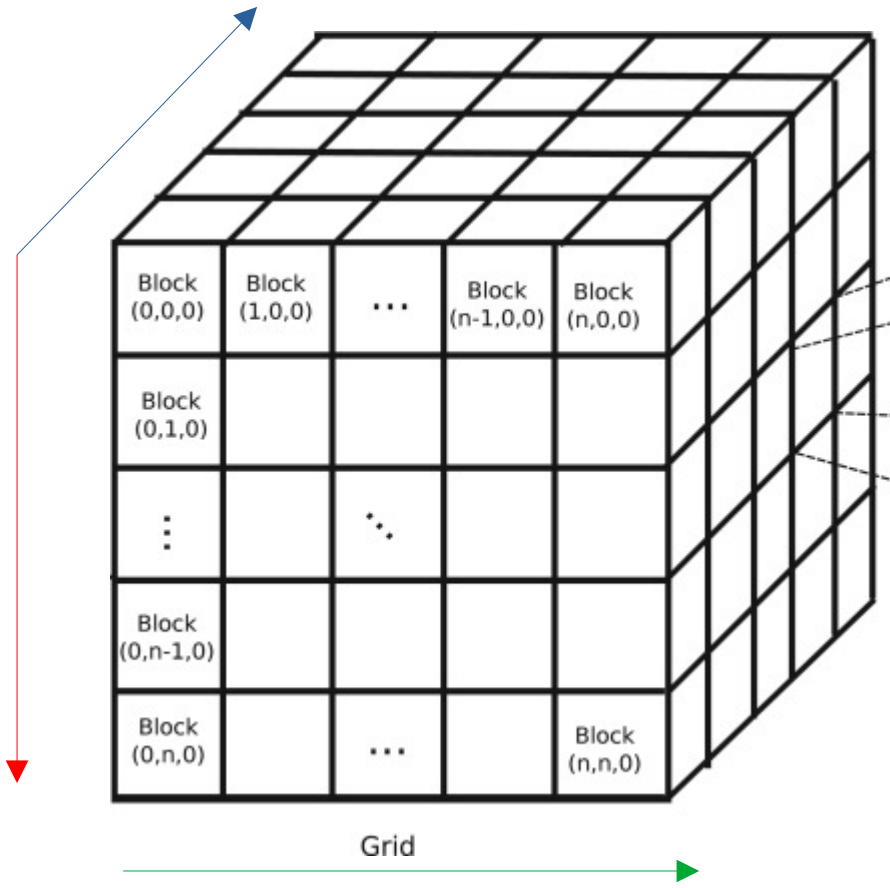


Q4

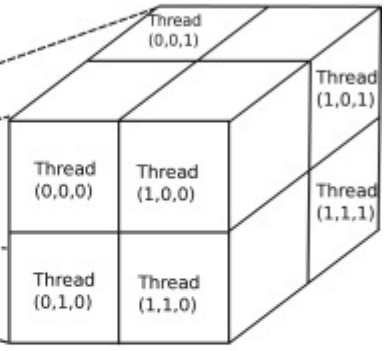



```
th = (threadIdx.z *
blockDim.y * blockDim.x) +
threadIdx.y * blockDim.x
+ threadIdx.x
```

Calcul d'indice global: grille 3D



Q5



indice_global =
block_id * size
+ th