



Introducción a CUDA

Miller Mendoza ^{1,2} y José Daniel Muñoz ¹

¹ Grupo Simulación de Sistemas Físicos, CEIBA-Complejidad
Departamento de Física, Universidad Nacional de Colombia,
Crr 30 # 45-03, Ed. 404, Of. 348, Bogotá D.C., Colombia

² ETH-Zürich, Computational Physics for Engineering Materials,
Institute for Building Materials,
Schafmattstrasse 6, HIF, CH-8093 Zürich, Switzerland



COLCIENCIAS



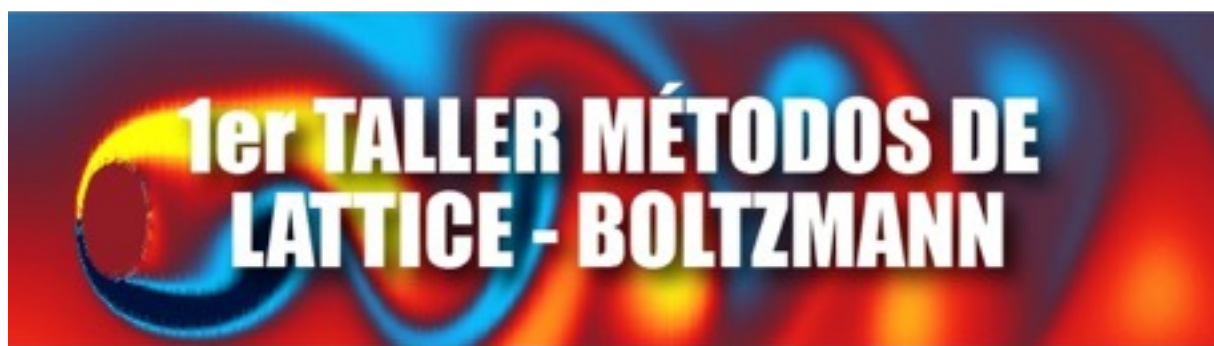
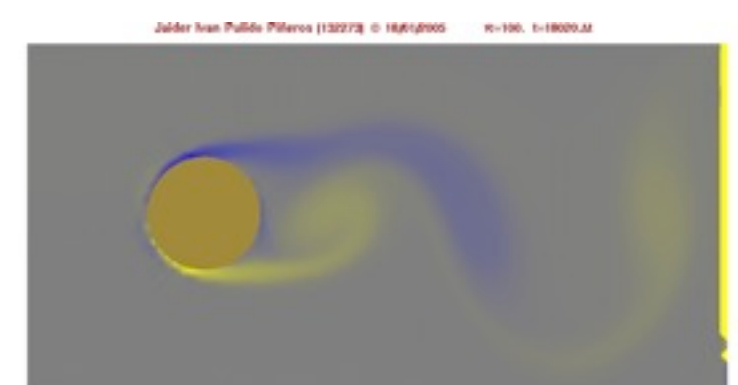
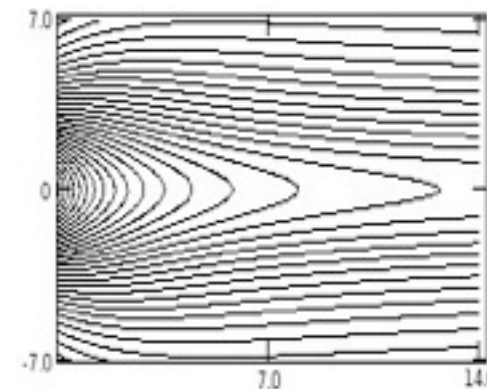
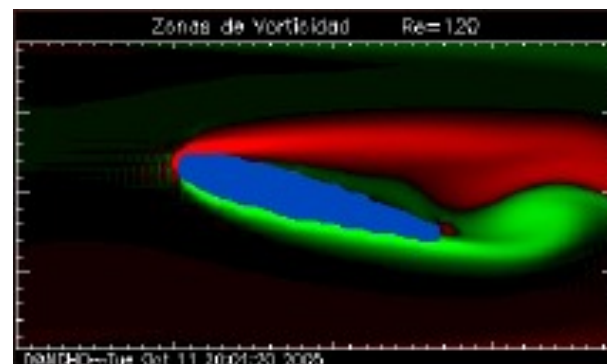
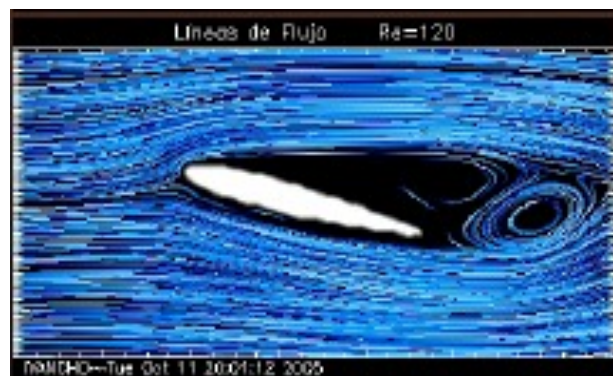
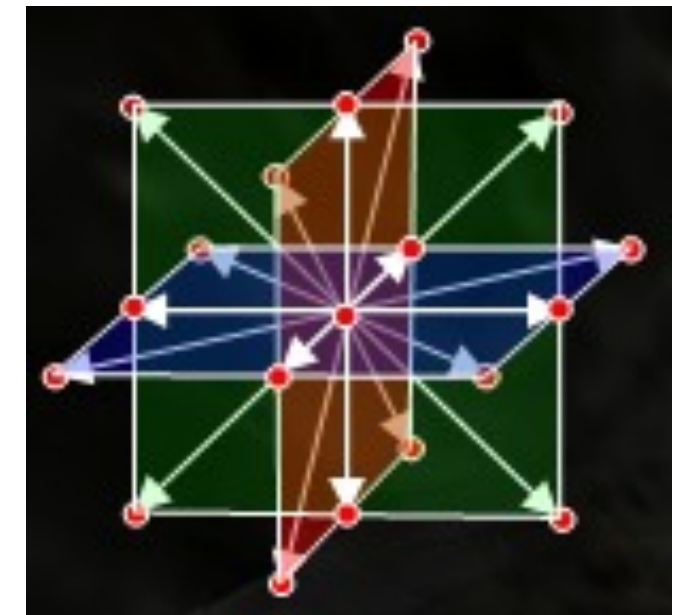
UNIVERSIDAD
NACIONAL
DE COLOMBIA

Introducción a CUDA

Miller Mendoza ^{1,2} y José Daniel Muñoz ¹

¹ Grupo Simulación de Sistemas Físicos, CEIBA-Complejidad
Departamento de Física, Universidad Nacional de Colombia,
Crr 30 # 45-03, Ed. 404, Of. 348, Bogotá D.C., Colombia

² ETH-Zürich, Computational Physics for Engineering Materials,
Institute for Building Materials,
Schafmattstrasse 6, HIF, CH-8093 Zürich, Switzerland

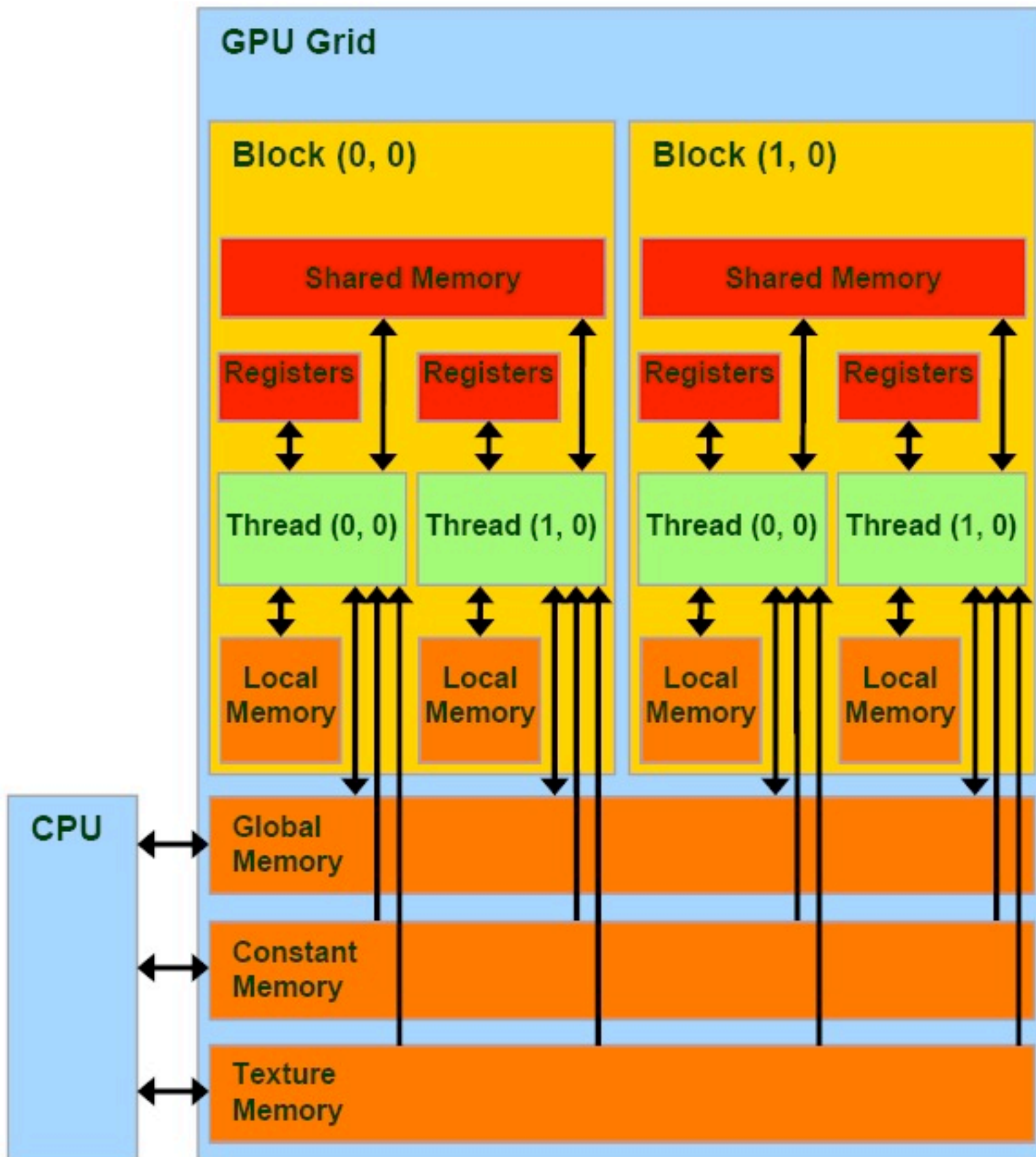


COLCIENCIAS



UNIVERSIDAD
NACIONAL
DE COLOMBIA

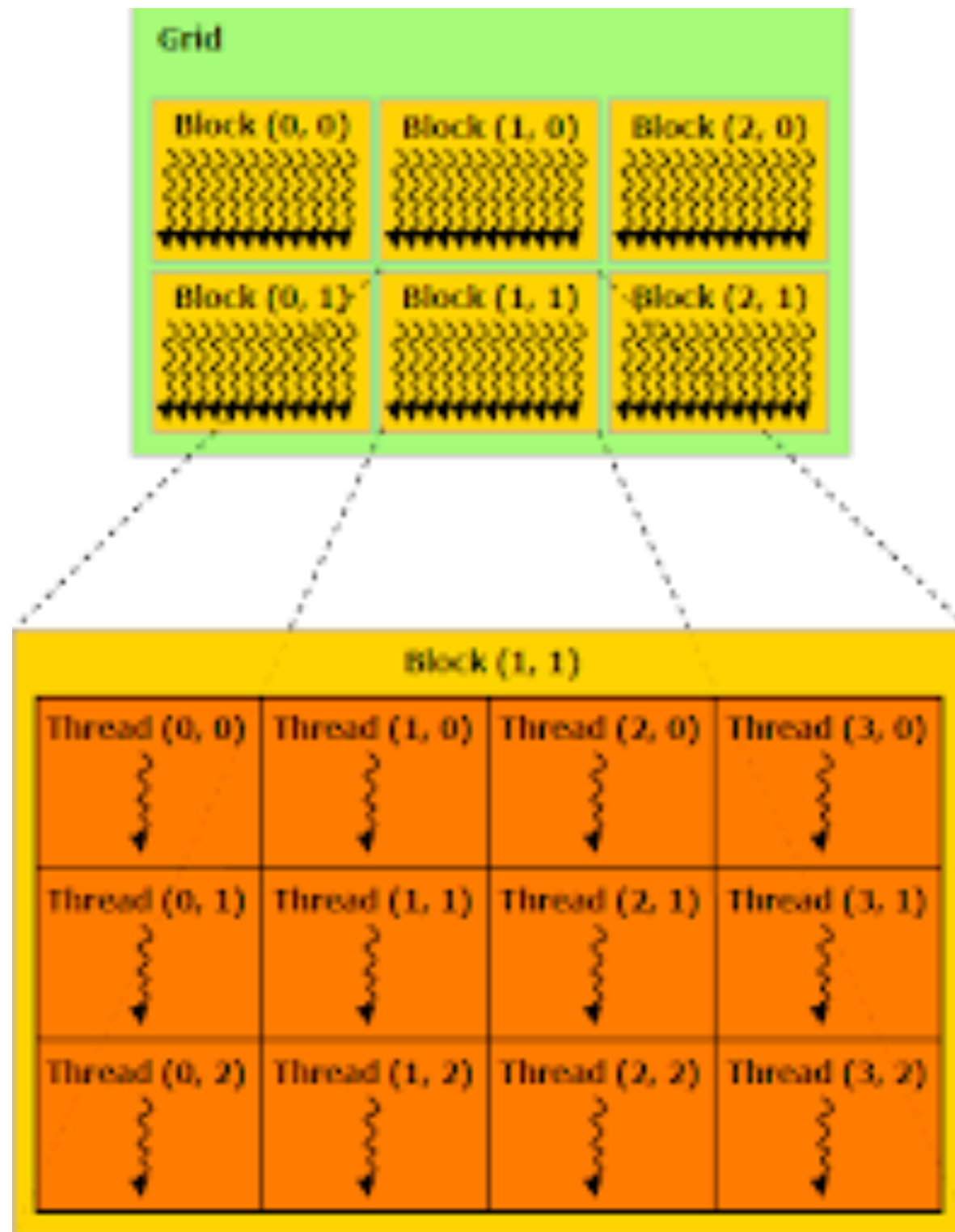
La tarjeta gráfica



Diferentes tipos de memoria

Entre más lejos, más grande, pero más lenta

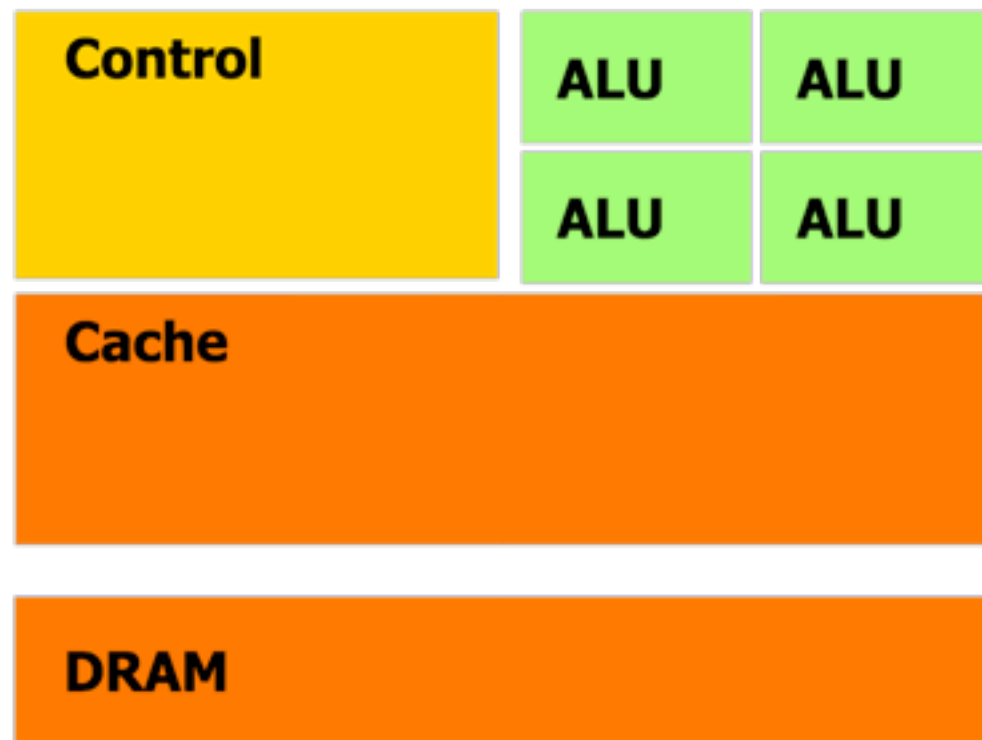
Grid, Blocks and Threads(hilos)



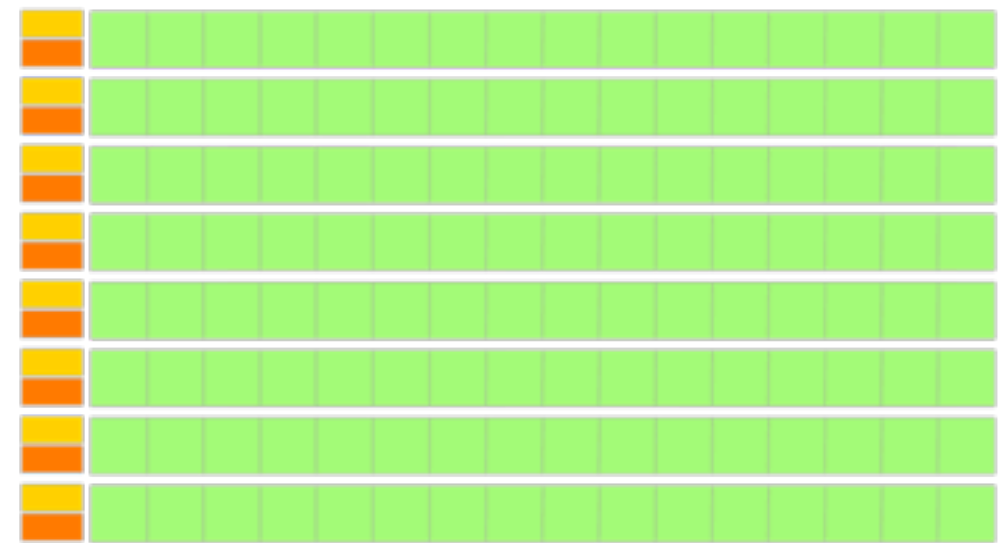
Las tareas se mandan en bloques de hilos

Cada hilo es lo que hace un procesador

Intercambio



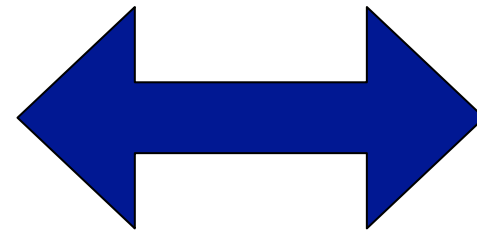
CPU



DRAM

GPU

Host



Device

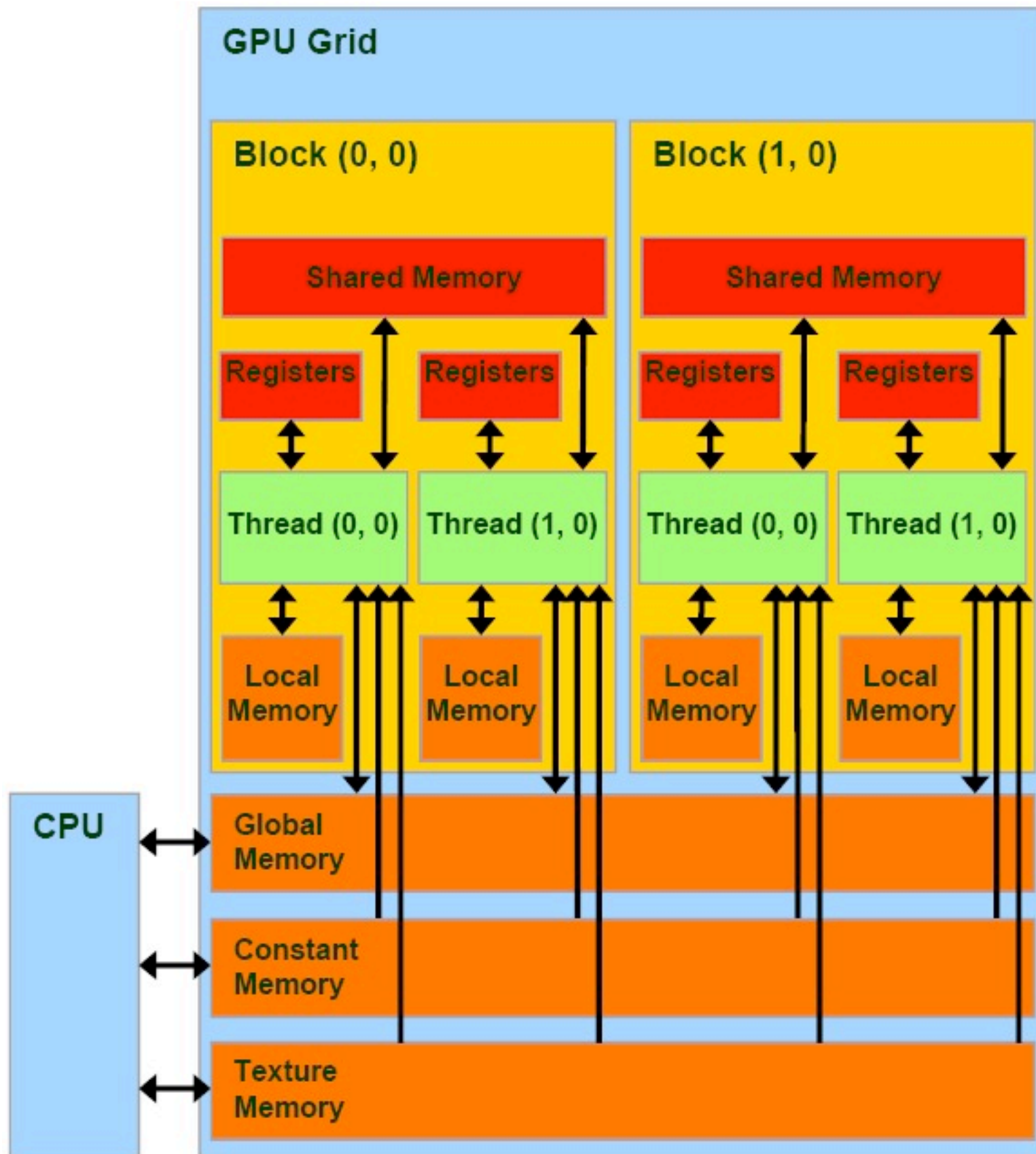
Se mantiene una copia de los datos en cada uno,
y se transfiere

**Kernel=algo para que
todos hagan.**

- Mi primer programa en CUDA
- Suma de dos vectores en CUDA

Constantes en CUDA

Diferentes tipos de memoria



Pueden ser
accesadas por todos
los procesadores

Funciones en CUDA

CUDA Function Declarations

	Executed on the:	Only callable from the:
<code>__device__ float DeviceFunc() ↗</code>	device	device
<code>__global__ void KernelFunc() ↗</code>	device	host
<code>__host__ float HostFunc() ↗</code>	host	host

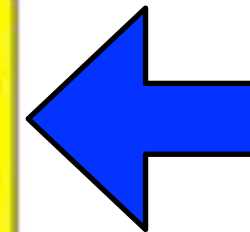
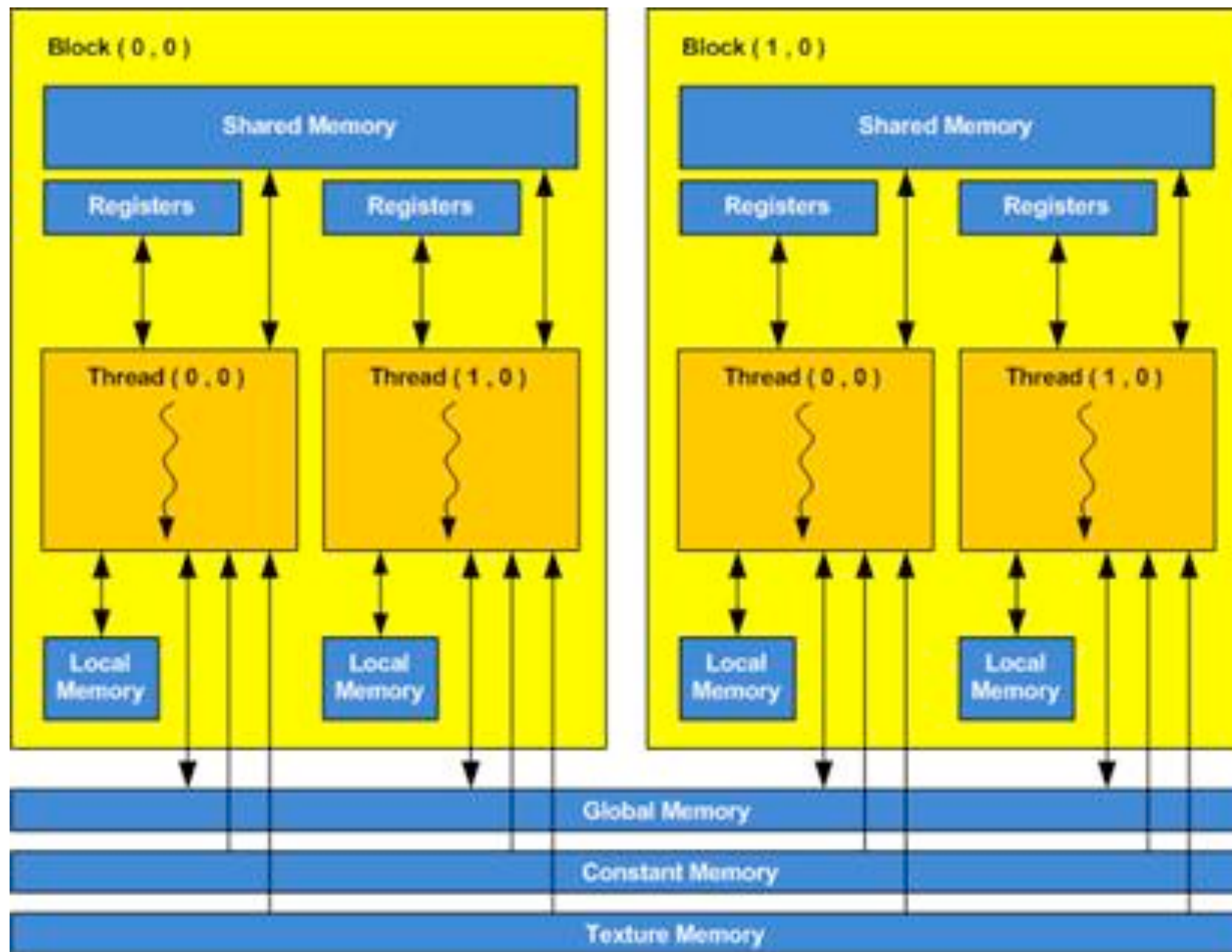
`__host__`

Es una orden a todos los procesadores
Debe ser `void`

`__device__`

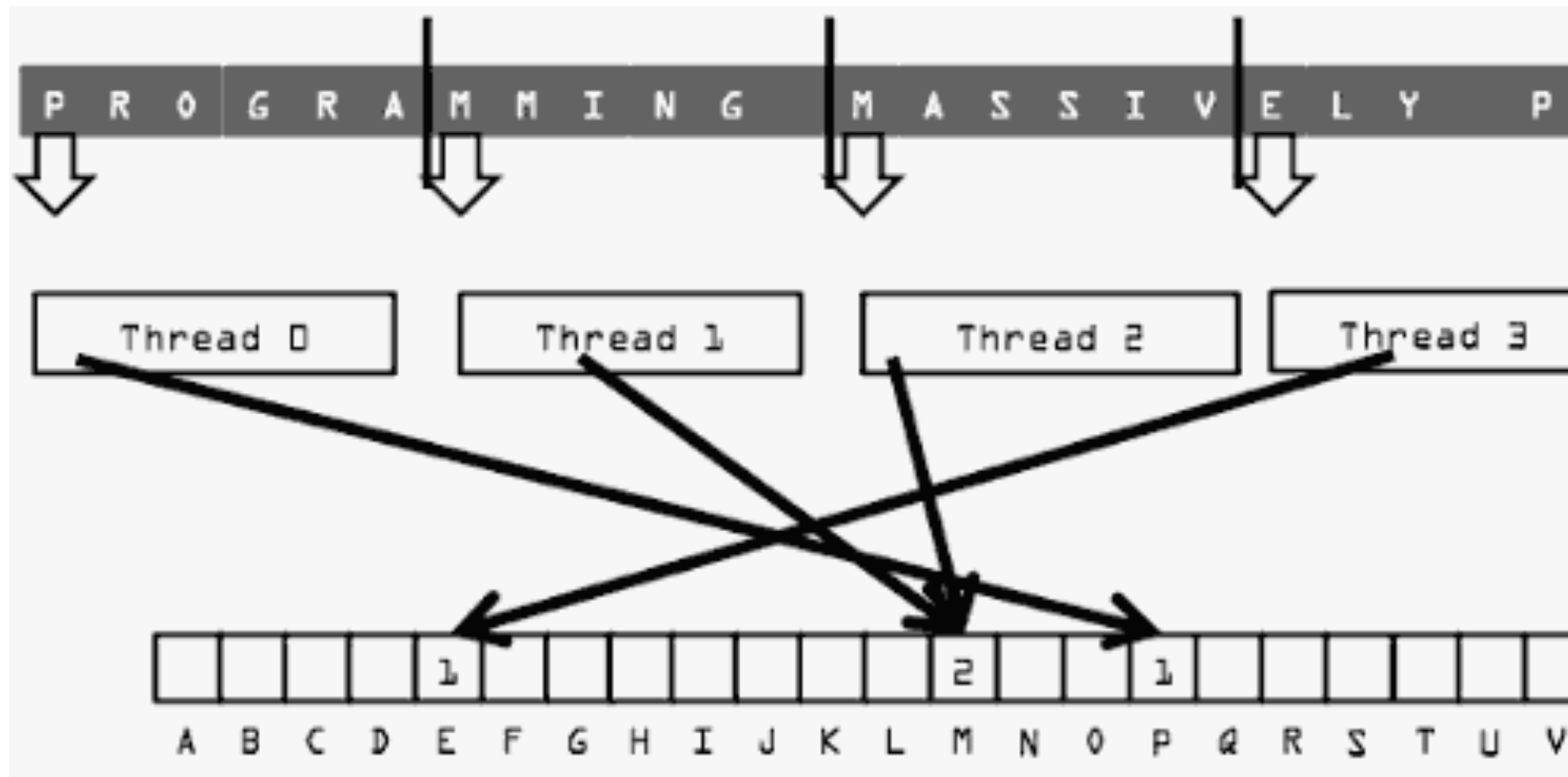
Es un cálculo parcial que hace cada procesador

Shared Memory en CUDA

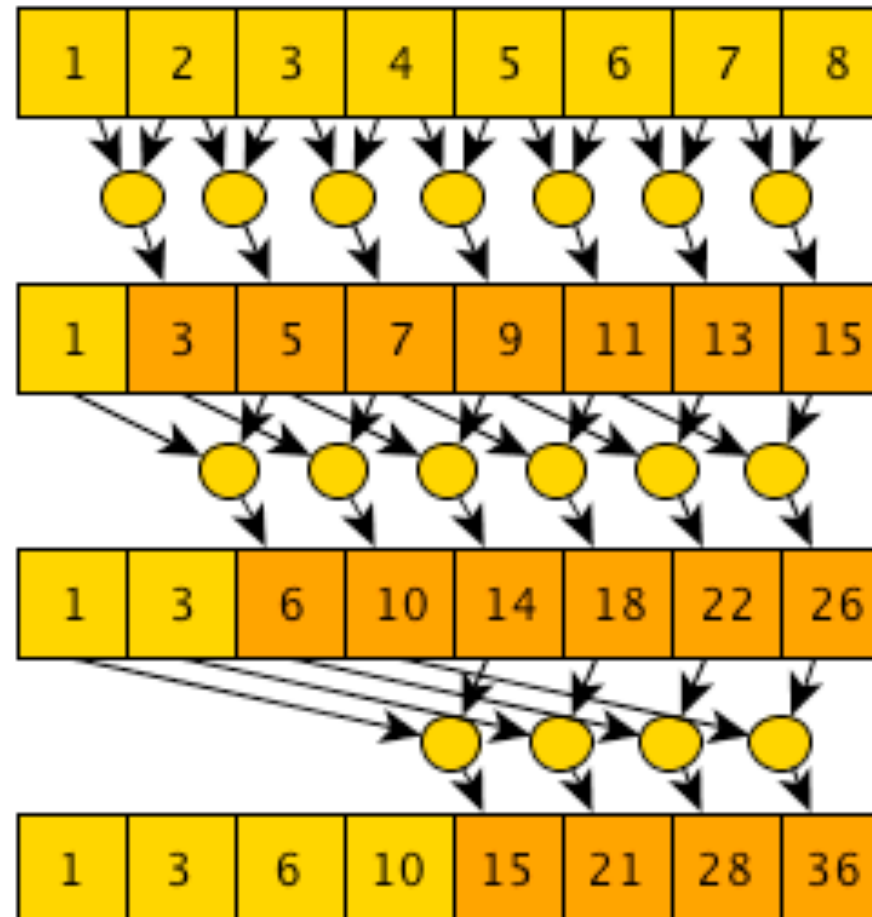


Es una memoria caché que todos los hilos de un mismo bloque pueden acceder

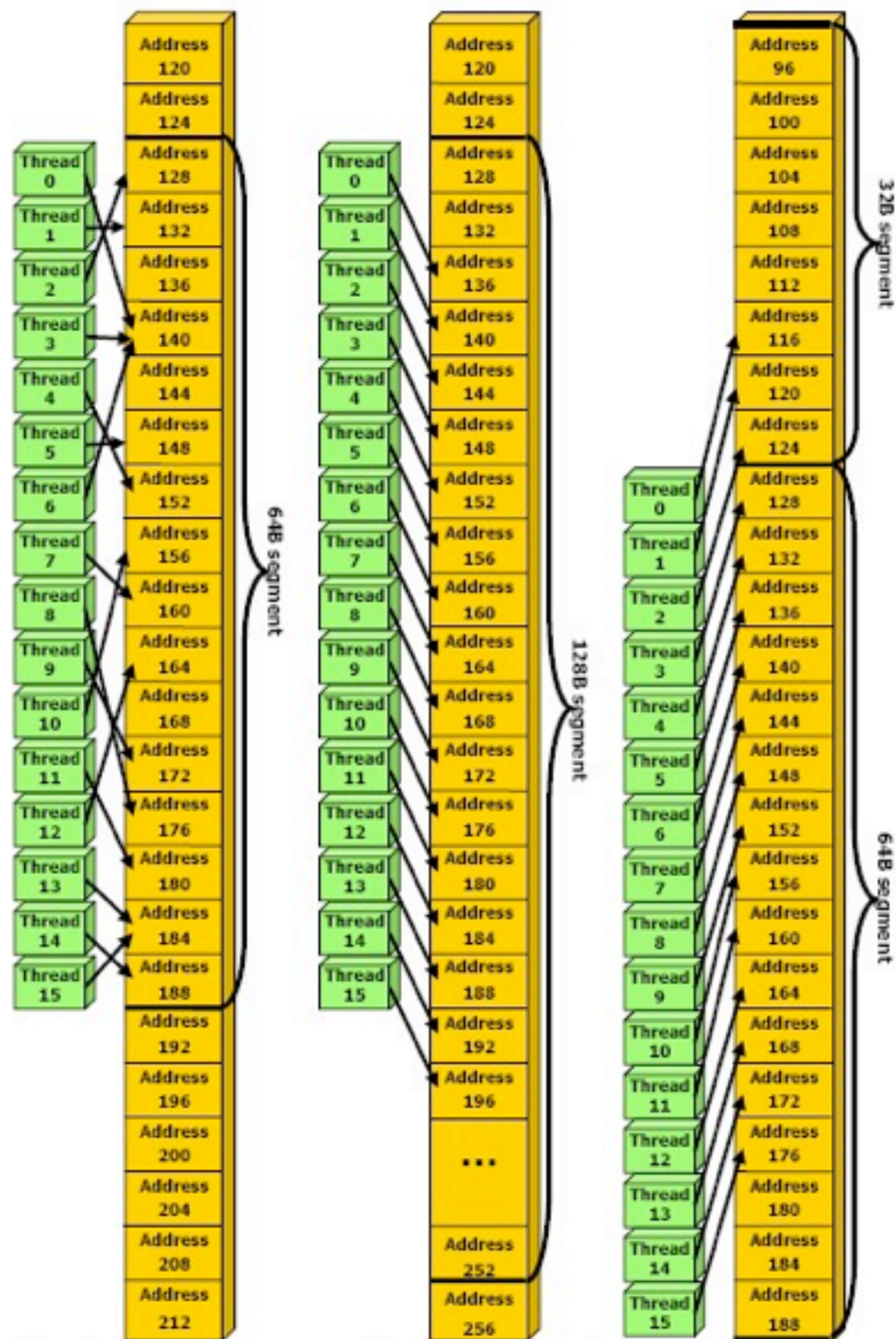
Atomic Operations in CUDA



Algoritmos en Paralelo



Matrices 2D y 3D: Pitch



Left: random float memory access within a 64B segment, resulting in one memory transaction.
Center: misaligned float memory access, resulting in one transaction.
Right: misaligned float memory access, resulting in two transactions.

La memoria se transfiere en bloques enteros, y eso corre el inicio de la siguiente columna de la matriz

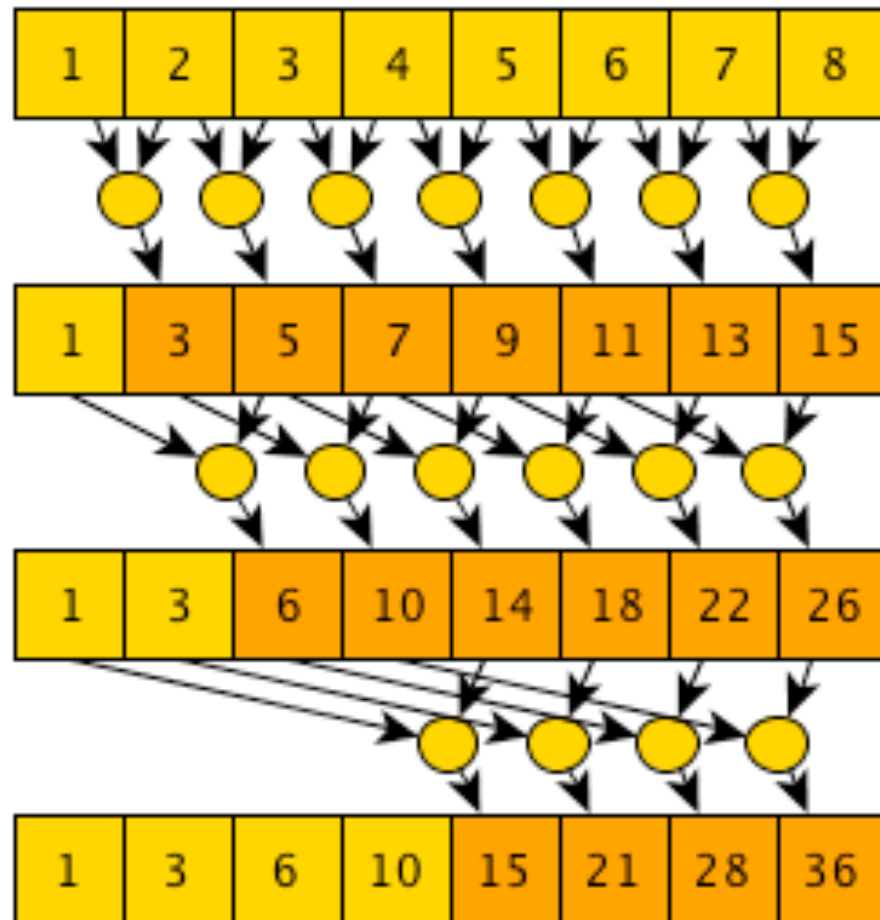
- Matriz 2D en CUDA

Reduce in CUDA

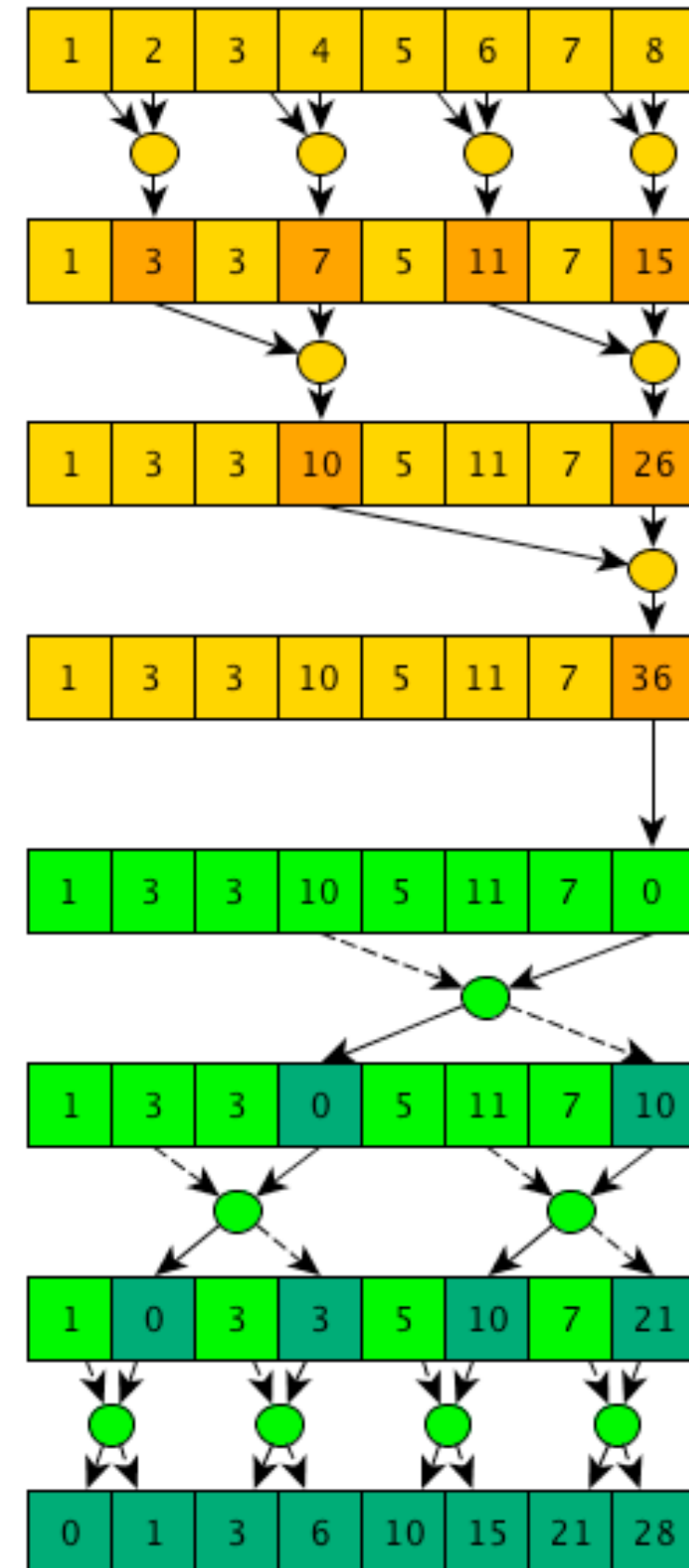


Scan

Hillis/Steele



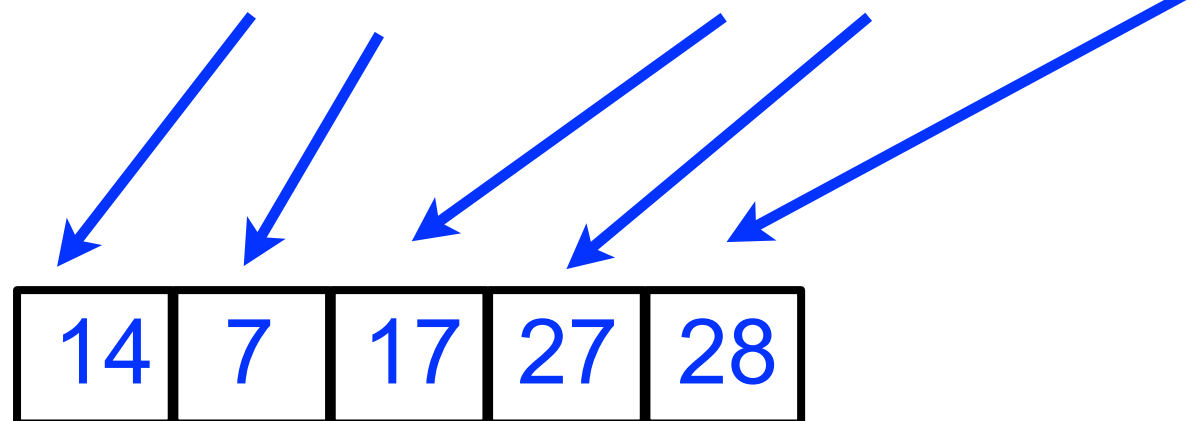
Block



Compact

Quiero seleccionar
sólo algunos

18	14	7	3	17	2	27	28	5
----	----	---	---	----	---	----	----	---



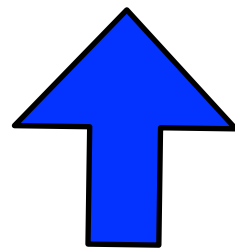
¿Dónde los escribo?

0 1 2 3 4

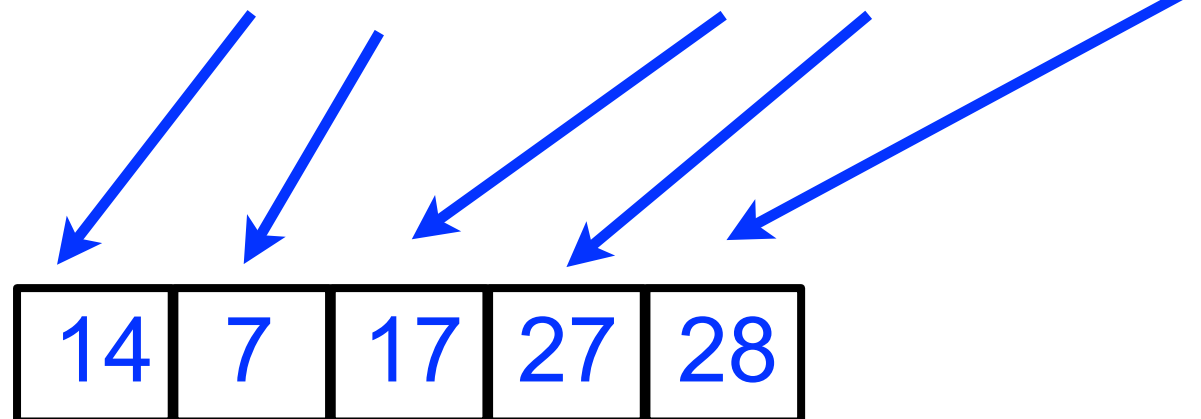
Compact

0	1	1	0	1	0	1	1	0
---	---	---	---	---	---	---	---	---

Quiero seleccionar
sólo algunos



18	14	7	3	17	2	27	28	5
----	----	---	---	----	---	----	----	---



¿Dónde los escribo?

14	7	17	27	28
----	---	----	----	----

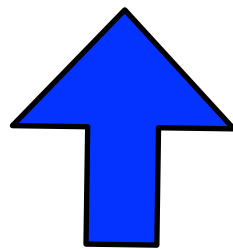
0 1 2 3 4

Compact

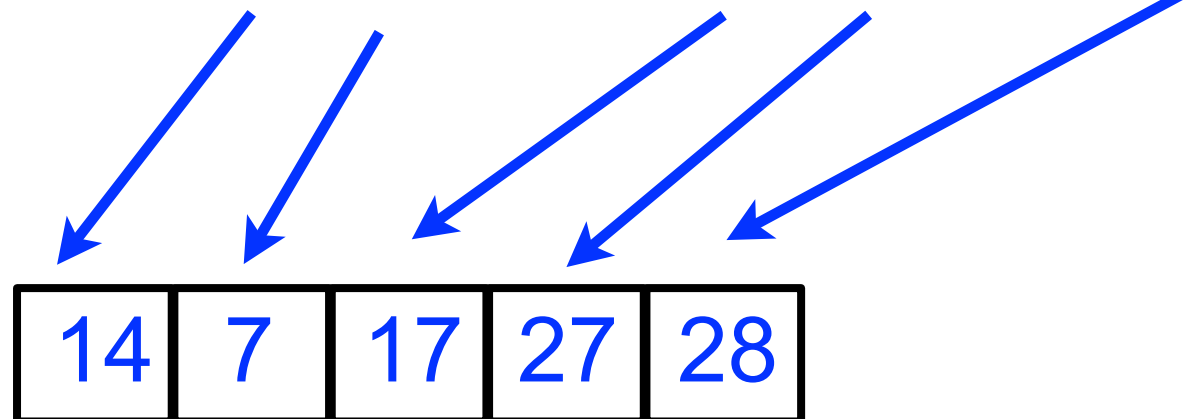
0	1	1	0	1	0	1	1	0
---	---	---	---	---	---	---	---	---

0 0 1 2 2 3 3 4 5

Quiero seleccionar
sólo algunos



18	14	7	3	17	2	27	28	5
----	----	---	---	----	---	----	----	---



¿Dónde los escribo?

14	7	17	27	28
----	---	----	----	----

0 1 2 3 4

Curso de CUDA en Udacity

The screenshot shows a web browser window displaying a Udacity classroom page. The browser's address bar shows the URL: <https://classroom.udacity.com/courses/cs344/lessons/86719951/concepts/880351220923>. The page title is "Hillis Steele Scan". On the left, a sidebar lists lessons, with "22. Quiz: Hillis Steele Scan" highlighted. The main content area features a video player showing a hand-drawn diagram of the Hillis Steele Inclusive Scan. The diagram consists of two rows of numbers: the top row has numbers 1 through 8, and the bottom row has numbers 1 through 15. Green lines connect the numbers in the top row to the numbers in the bottom row, illustrating the cumulative sum process. The video player controls at the bottom show a progress bar at 0:58 / 2:14 and the YouTube logo.

<https://in.udacity.com/course/intro-to-parallel-programming--cs344>

Gracias!

jdmunozc@unal.edu.co