Universidad Católica San Pablo

CUDA **Introducción**

Marc-Antoine Le Guen

mleguen@ucsp.edu.pe

07/07/2017

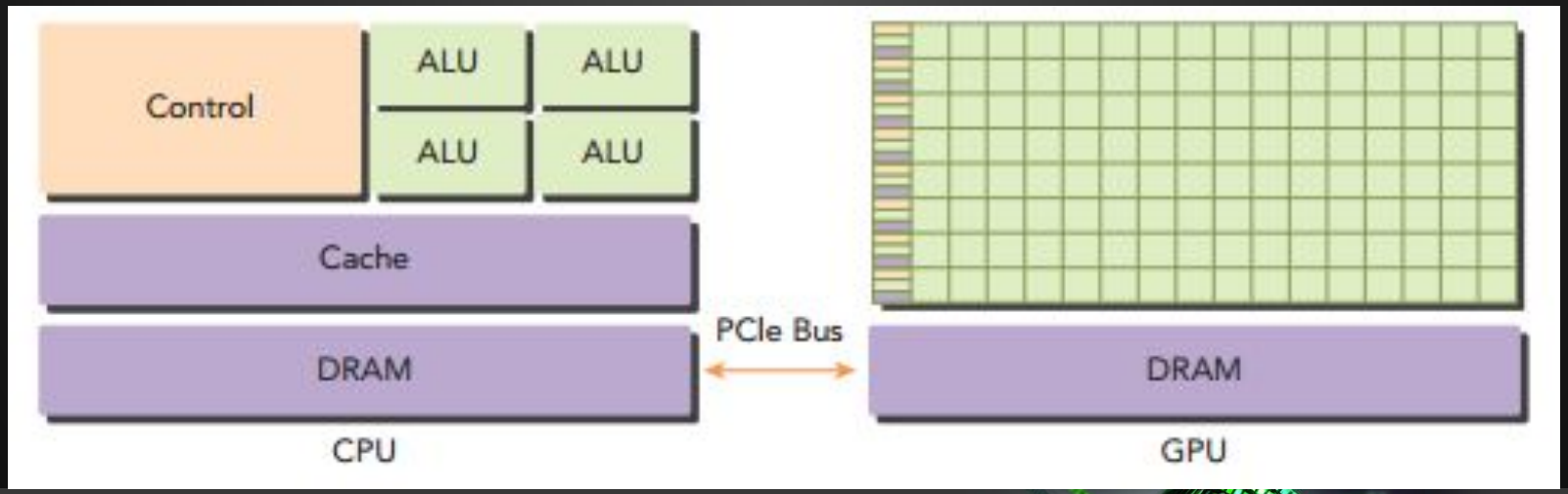
GPGPU

- **General-purpose computing on graphics processing units**
  - Realizar tareas en el GPU generalmente realizadas en el CPU

HPC

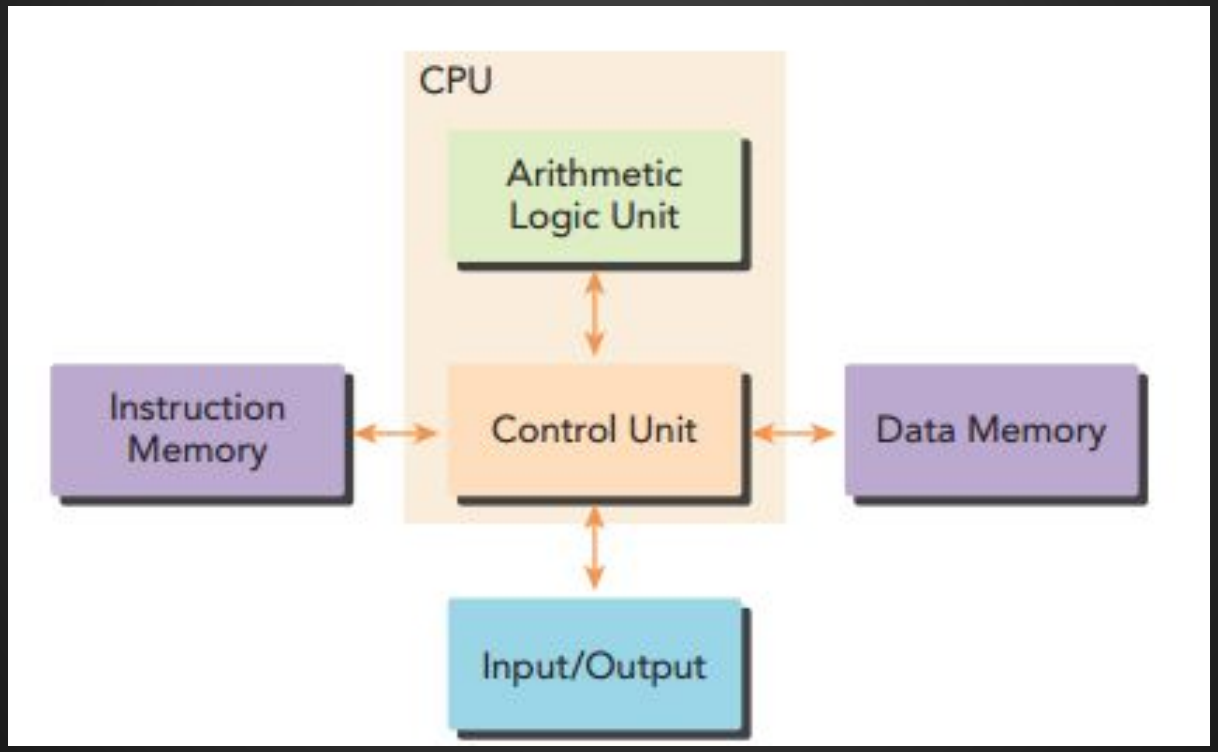
High Performance Computing

- **Arquitectura heterogénea CPU-GPU**
- **Computación paralela consiste en llevar a cabo cálculos al mismo tiempo**
  - Arquitectura de la computadora
  - Programación paralela

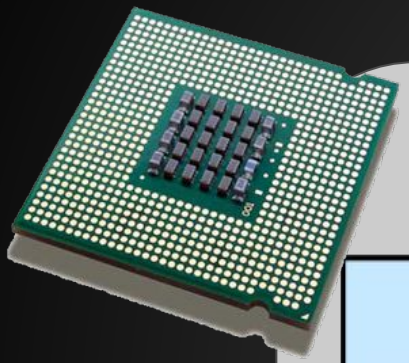


Central Processing Unit

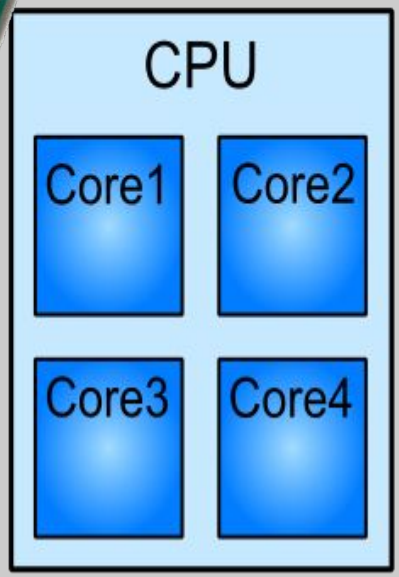
- **Múltiples núcleos para soportar el paralelismo**

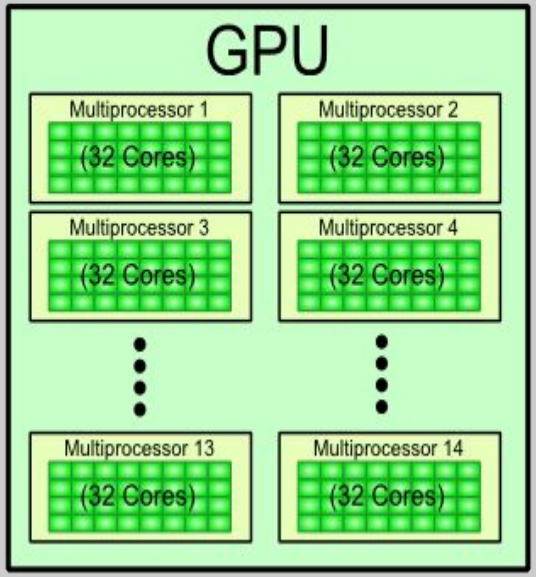


Arquitectura



CPU/GPU Architecture Comparison

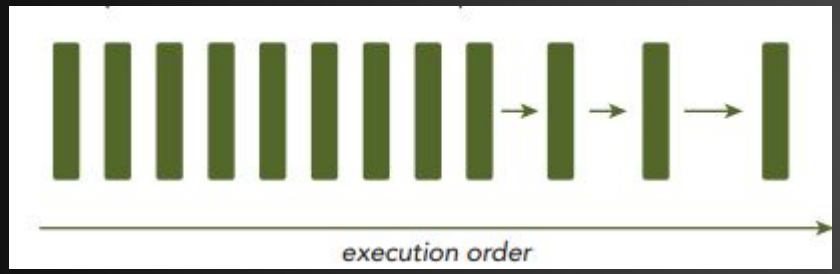


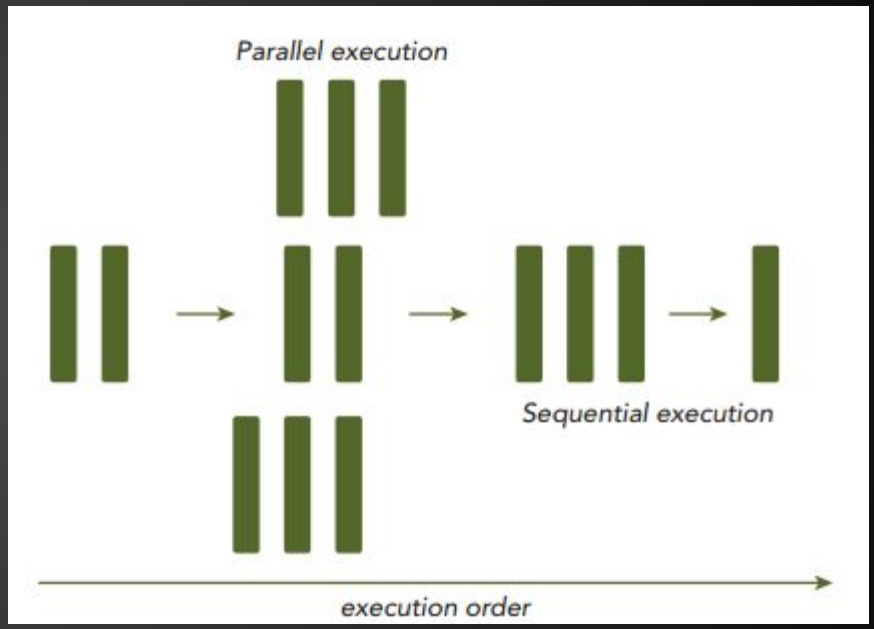


Sistema many-core

Programa

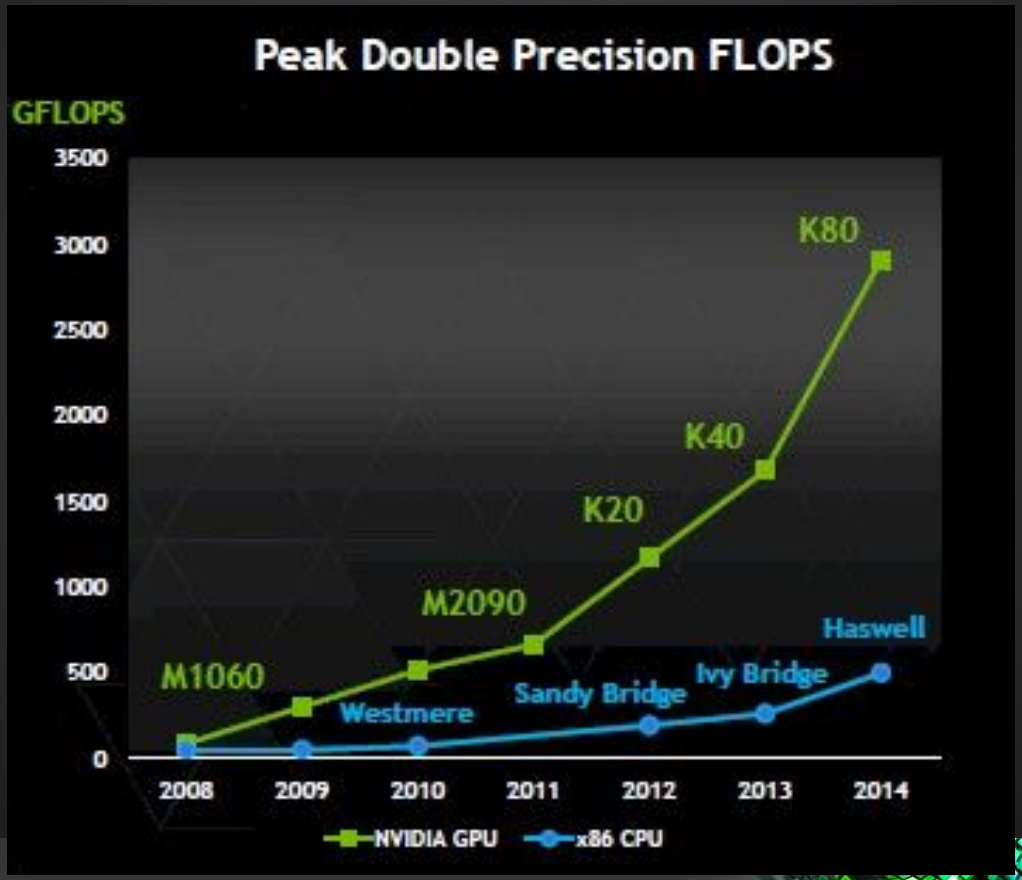
- **Tareas simultáneas (paralelizable)**
- **Tareas secuenciales**
- **Analizar la dependencia de los datos**





Procesador masivamente paralelo

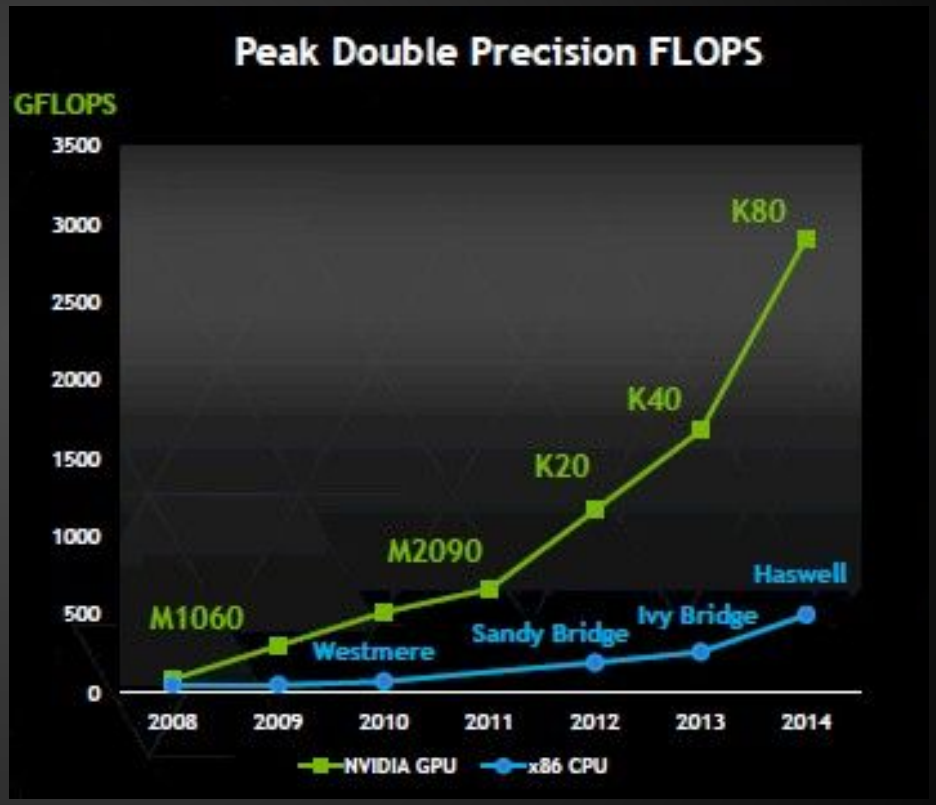
- Gflops (operación a coma flotante por segundo)



Procesador masivamente paralelo

- Memory **Bandwidth**

- 1080 GTX
  - 9 Tflops
  - Memory Width 256-bit
  - Memory BW 320 GB/s
- Intel i7 5960x
  - 3.84 Tflops
  - Memory BW 68GB/s



Programa

- **Data parallelism : distribuir los datos sobre varios núcleos**
- **Task parallelism : distribuir diferentes tareas a varios núcleos**

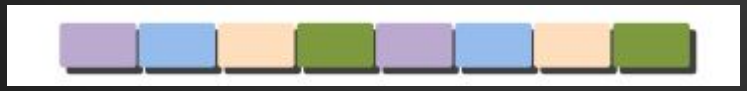
La programación en GPGPU es adaptada a la paralelización de datos donde cada thread trabajara sobre una pequeña porción de los datos (chunk).

Partición por bloque

- **Los bloques son varios chunks juntos**

Partición cíclica

- **Un thread maneja pequeños chunks, un nuevo chunk para un thread necesitará un salto de N_thread chunks**

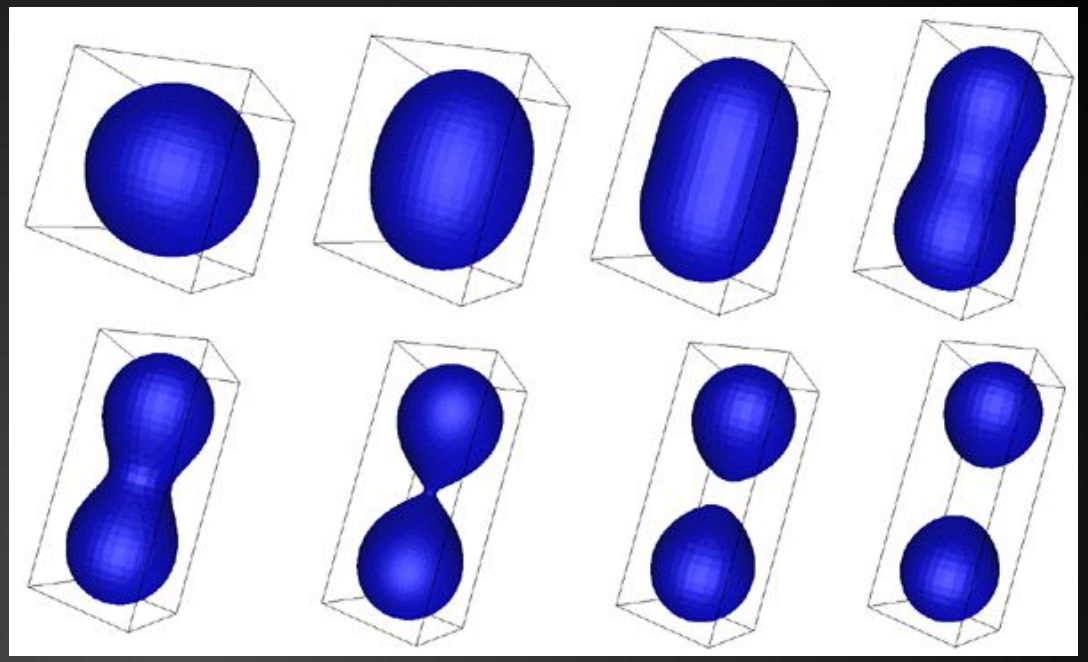


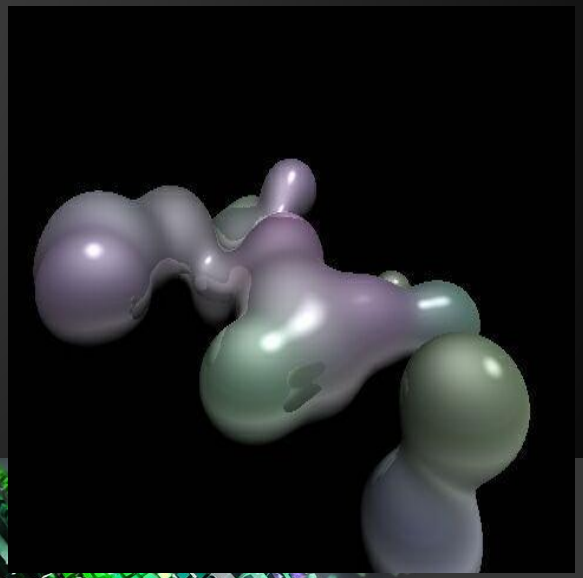
Inicio de GPGPU

- Shaders
  - Input
    - Atributos de un vértice
    - Color
    - Textura
    - Coordenadas de texturas
    - Atributos arbitrarios
  - Resultado
    - propiedades de un vértice
    - Color del pixel (fragmento)

Ejemplo

- Iso Surface
  - Semillas
  - Grid 3D (Discretización)
- Inputs
  - Coordenadas de semillas
  - Texture
    - 1 plano del grid 3D
  - O Malla 3D
- Output
  - Texture 0 = dentro/1 = fuera
  - O Malla 3D editada

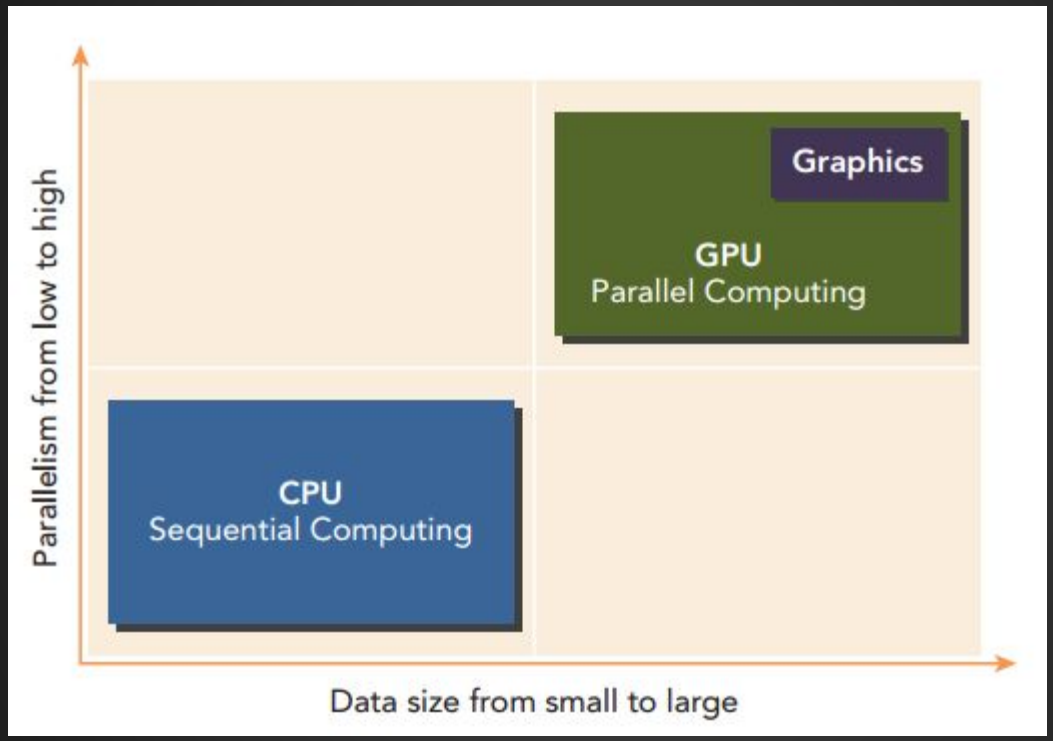




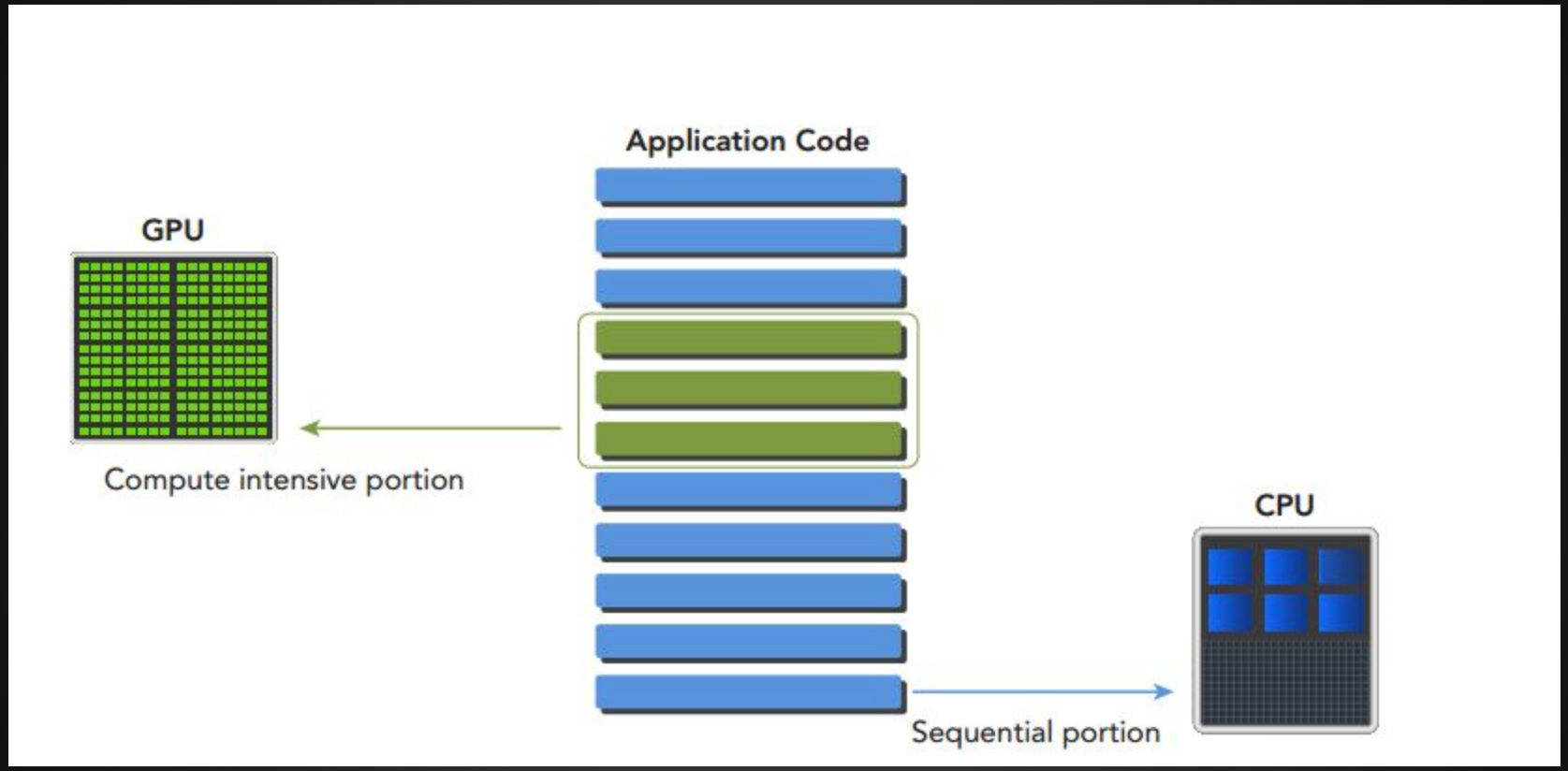
Para qué usar GPGPU ?

- Procesamiento de imágenes
- Simulación física
- Cálculos científicos
- Generación de imágenes
- Cualquier algoritmo paralelo

Cuando usar GPGPU ?



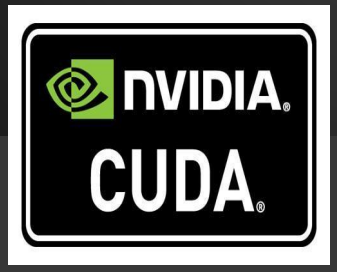
Cómo usar GPGPU ?

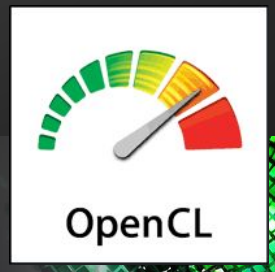


GPGPU

- CUDA, **lenguaje** de alto nivel, Nvidia
  - GPU nvidia
  - versión 9.1

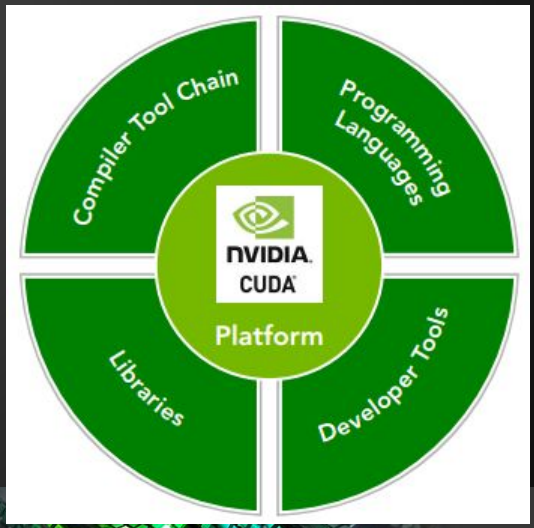
- OpenCL, **lenguaje** de alto nivel, libre, « starter group » AMD-ATI, Apple e Intel, las especificaciones son hechas por el Khronos Group
  - Cualquier procesador
  - versión 2.2





CUDA

- Standard C
  - Keywords
  - Compilador
- 2007 - Nació el primer lenguaje dedicado al GPGPU. (GTX 8800)



CUDA

Compatibile con productos de nvidia

- **Tegra**
- **Geforce**
- **Tesla**
- **Quadro**

Características:

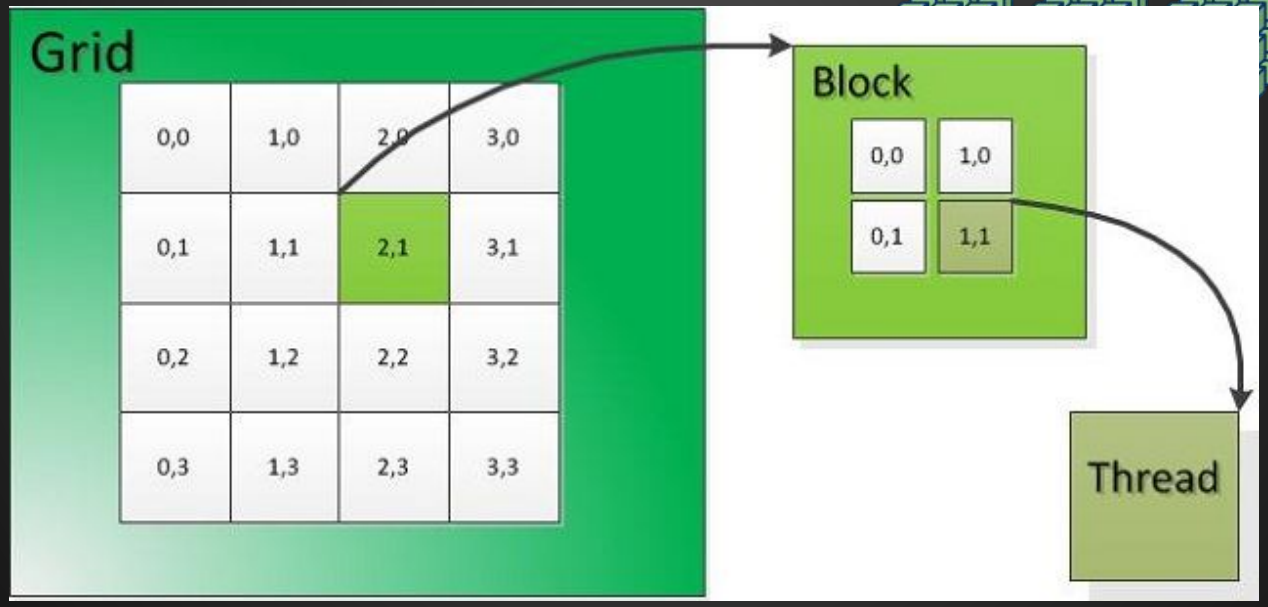
- **Tamaño en memoria**
- **Cantidad de cuda cores**

Métricas:

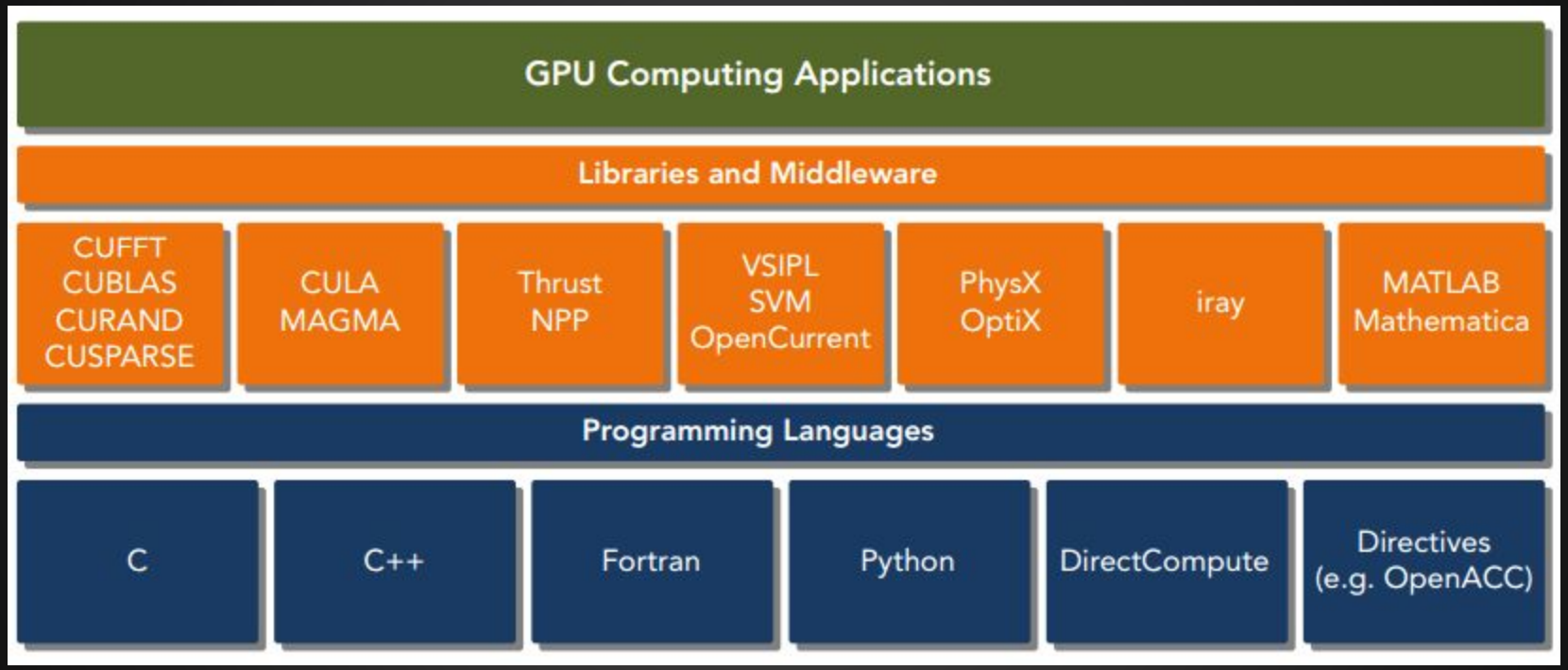
- **Capacidad computacional (tflops)**
- **Ancho de banda de memoria (Gb/s)**

Arquitectura **CUDA**

- Grid
  - Blocks
    - Threads



Arquitectura CUDA



Cuda - Set Up

- **Cross platform**
  - Windows - Visual Studio (Camino fácil)
  - Mac OSX 10.9 +
  - Linux
- **https://developer.nvidia.com/cuda-downloads**
- **http://docs.nvidia.com/cuda/index.html#axzz4CMbQFAp3**

OpenGL

- **OpenGL es :**
  - **Open G**raphics **L**ibrary
  - API (Application Programming Interface)
  - Multi Platform
  - Rendering 2D/3D
  - Interactúa con el GPU
- **OpenGL NO es :**
  - sistema de creación de ventana
  - sistema de interfaz de usuario
  - no contiene el framebuffer
    - gestionado por el sistema de ventana (GLUT/Qt/GLFW)