NVIDIA CUDA

Introducción y Ejemplos.

Jonathan Antognini C.

Universidad Técnica Federico Santa María

20 de agosto de 2012

Introducción

- Compute Unified Device Architecture (CUDA), es una tecnología desarrollada por NVIDIA Corporation en el 2007.
- Soportado de la serie G8X en adelante.
- Compilador (nvcc) + conjunto de herramientas de desarrollo en C/C++.
- Existen wrappers para otros lenguajes como: Python, Fortran, Java.
- El SDK está disponible para Linux, Windows y Mac.

Nvidia Cuda

¿Por qué CUDA?

Se intenta explotar las ventajas de las GPU utilizando paralelismo soportado por los múltiples núcleos de una tarjeta gráfica.
Las GPU NVIDIA son arreglos de multiprocesadores, cada uno de los cuales tiene:

- Varios cores, que ejecutan el mismo programa concurrentemente.
- Memoria compartida, y mecanismo de sincronización.

Empezando en Cuda

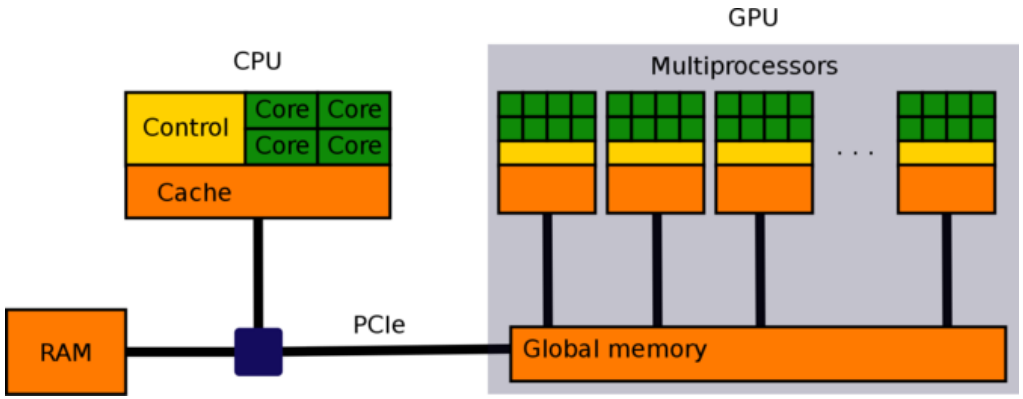
Cuda Downloads: Última versión estable: 4.2. Hay un RC 5.0
`http://developer.nvidia.com/cuda/cuda-downloads`

- Instalar toolkit.
- Instalar driver compatible.
- Instalar SDK.

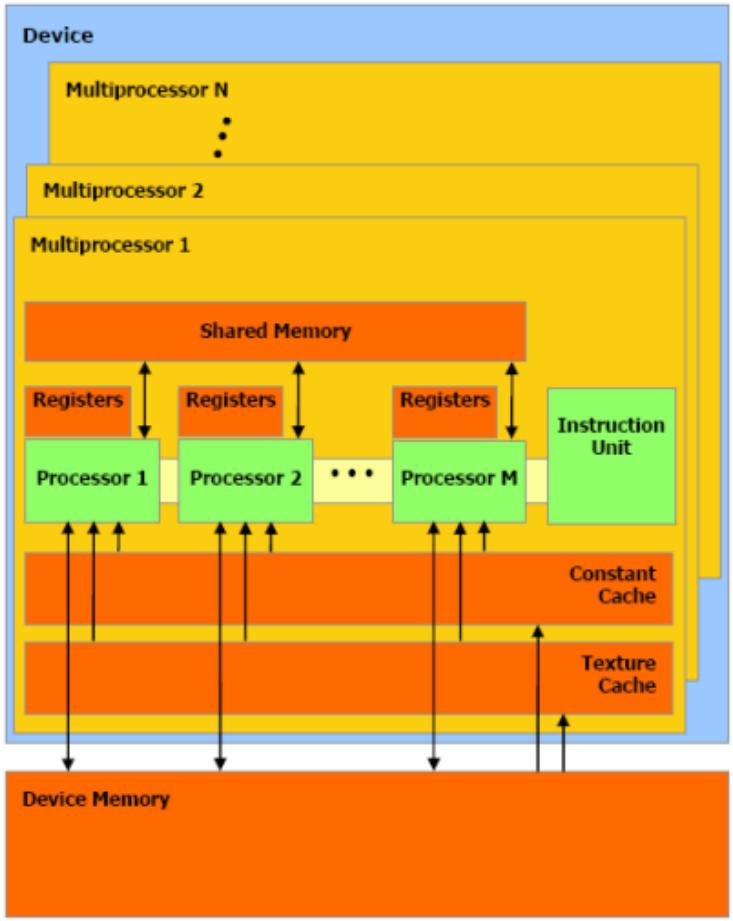
Cuda library documentation:
`http://www.clear.rice.edu/comp422/resources/cuda/html/index.html`

Arquitectura de Cuda

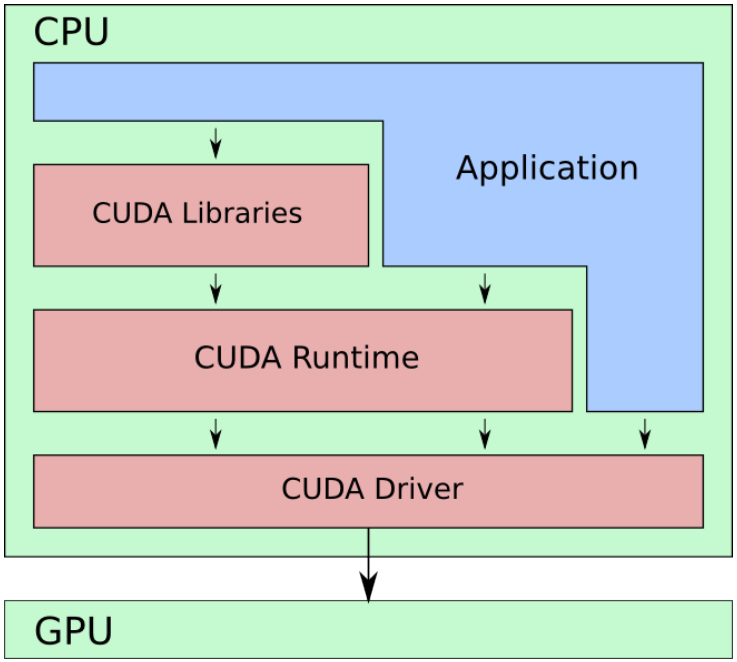
Para efectos de hardware, la GPU funciona como coprocesador, pudiéndose utilizar de forma simultánea a la CPU y así aprovechar el potencial que puedan ofrecer ambas al mismo tiempo. Esta arquitectura de procesamiento paralelo masivo es la que proporciona a la GPU su alta capacidad de cálculo.



Arquitectura de Cuda



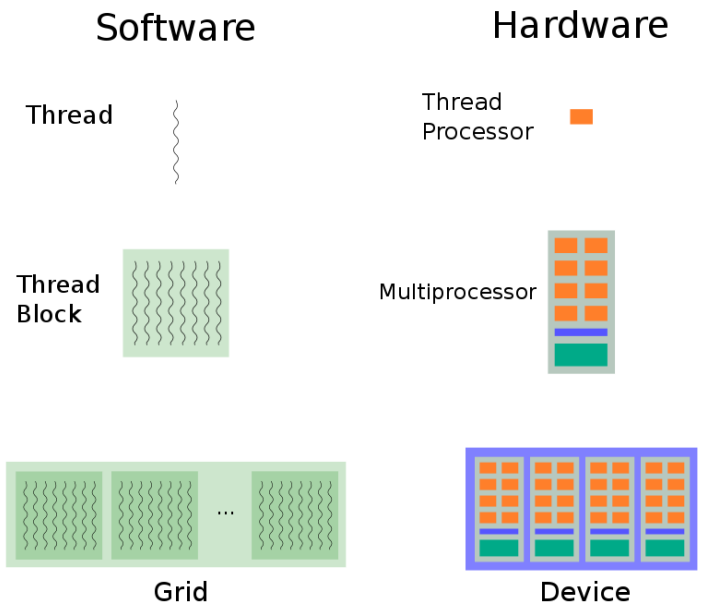
Comunicación entre host y device



Hebras y Bloques

- Cada hebra se ejecuta en un Streaming processor.
- Un bloque es una agrupación fija de hebras.
- Cada bloque se ejecuta en un solo Streaming Multiprocessor.
- Un SM puede tener asignados varios bloques.
- Kernel: función invocada desde la CPU que se ejecuta en la GPU. Recibe como parámetro estructuras que definen la cantidad de bloques y la cantidad de hebras por bloque.

Hebras y Bloques



Estructura de un programa en CUDA

Por lo general, la estructura de un programa en CUDA tiene esta secuencia:

- Generar datos en la CPU.
- Copiar datos de la CPU a la GPU.
- Realizar cálculo en la GPU.
- Copiar los resultados de la GPU a la CPU.

Ejemplos

- Suma de vectores.
- Histograma.

Conclusiones

- CUDA permite sacar provecho a las potencialidades de las GPU's actuales.
- Abordar problemas paralelos es más sencillo.
- Sin embargo es fácil programar en CUDA, pero es difícil conseguir rendimientos elevados.

EOF