UNIVERSIDAD AUTONOMA DE MADRID

Procesamiento de Grandes Volumenes de Datos

Programación GPGPU

1. Recursos de la GPU **3**

2. Suma de 2 matrices **4**

3. Stencil1d: Estudiar el efecto de la memoria compartida **5**

1. Recursos de la GPU

En un nuevo cuaderno de Google Colab

Ir al enlace https://colab.research.google.com en un Navegador y haga Click en Nuevo Cuaderno

Seleccione la utilización de un coprocesador GPU

Click to Runtime > Change > Hardware Accelerator GPU .

Compruebe las características del sistema que le han proporcionado

```
!lscpu
```

```
!free -h
```

Verifique la versión de CUDA instalada

```
!nvcc --version
```

```
!nvidia-smi
```

Compruebe el directorio actual de trabajo

```
!pwd
!ls -la
!ls /
```

Compruebe las características de la GPU

```
%cd /usr/local/cuda/samples/1_Utilities/deviceQuery/
%ls
!make
```

```
!./deviceQuery
```

2. Suma de 2 matrices

El código de ejemplo en suma de los elementos de un vector realiza la suma de dos vectores en la GPU.

1. Comente loss diferentes casos propuestos en el ejemplo y conteste a las preguntas.

2. Se propone extender el código de este ejemplo para que realice la resta (o suma) de matrices cuadradas de dimensión N.
   - Configure adecuadamente el Grid de threads para aceptar matrices de cualquier tamaño.
   - En el kernel, utilice las variables blockIdx y threadIdx adecuadamente para acceder a una estructura bidimensional.

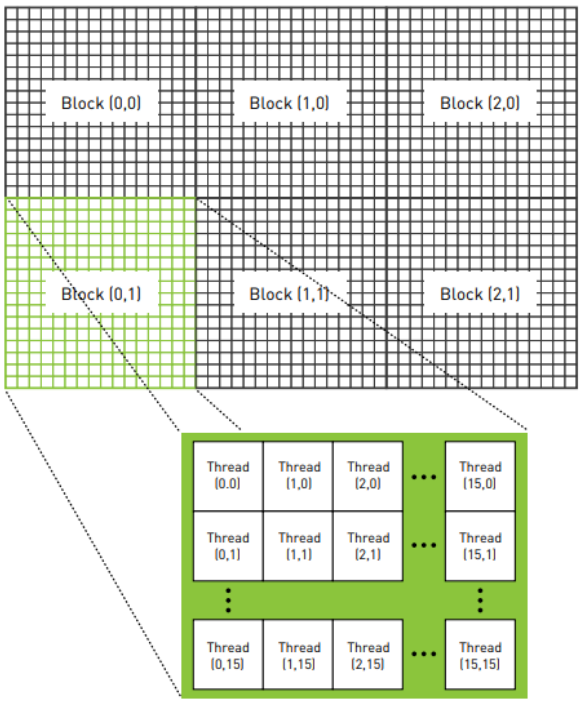

Figure 5.2 A 2D hierarchy of blocks and threads that could be used to process a 48 x 32 pixel image using one thread per pixel

Refs:

http://www.mat.unimi.it/users/sansotte/cuda/CUDA_by_Example.pdf

3. Stencil1d: Estudiar el efecto de la memoria compartida

Descripción:

El código stencil 1D es útil para entender los beneficios y uso de memoria compartida en una GPU. Para hacerlo explicito conviene valorar los resultados utilizando el generador de perfiles (nvprof) que muestra los cuellos de botella y su efecto en la aceleración final que se consigue.

Pasos:

1. Primero compile y ejecute el código sin usar memoria compartida. Hacer un perfil con nvprof y sacar el comportamiento temporal.

2. Use el generador de perfiles para determinar cuál es el problema.

3. Introduzca la modificación en el código para hacer uso de la memoria compartida.

Valore la situación que sucede cuando no se usa la función __syncthreads (). Ejecute el código varias veces y observe cuando se obtienen errores semi-aleatorios en la salida.

4. Añadiendo __syncthreads () evalue después de las diferentes modificaciones la aceleración obtenida.