FUNDAMENTOS DE PROGRAMACIÓN EN GPUS DE PROPÓSITO GENERAL CON CUDA

LABORATORIO DE INTEL PARA LA ACADEMIA
INSTRUCTORES
LAURA SANDOVAL MONTAÑO
ELBA KAREN SÁENZ GARCÍA
ARIEL ULLOA TREJO

TEMARIO

Antecedentes

GPU – CUDA
- GPGPU
- GPU Genérico
- Arquitectura CUDA

Modelo de ejecución CUDA
- Grids ,Bloques, Hilos

Modelo de programación CUDA
- Device y Host
- Funciones CUDA

Manejo de arreglos unidimensionales

Manejo de arreglos bidimensionales

Memoria Compartida

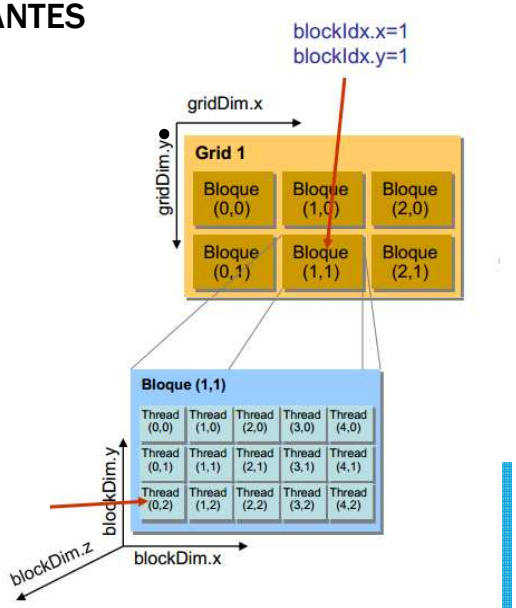
VARIABLES IMPORTANTES

gridDim : Dimensión de la grid .Número de bloques por grid.

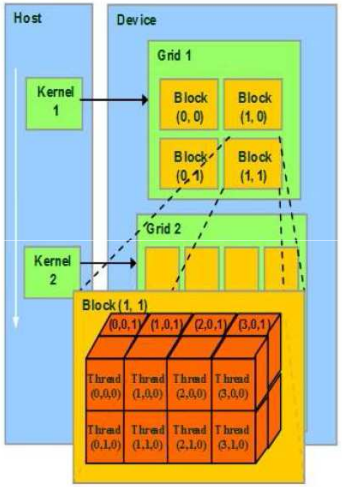
blockDim :Dimensión del bloque, numero de hilos por bloque.

threadIdx: id propio del hilo dentro del bloque al que pertenece

blockIdx: id de un bloque dentro de un grid



Dependiendo de la dimensión
- threadIdx (uint3) se accede con .x/.y/.z
- blockIdx (uint3) se accede con .x/.y/.z
- blockDim (dim3) se accede con .x/.y/.z
- gridDim (dim3) se accede con .x/.y/.z



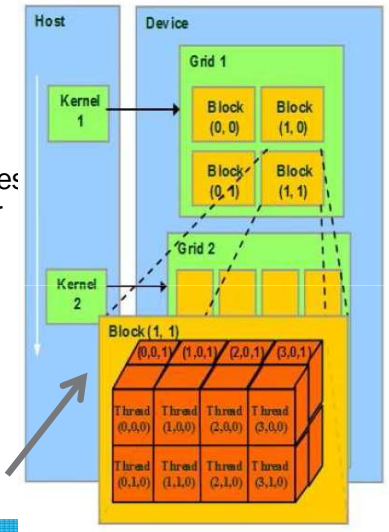
IMPORTANTE

- El número de hilos que es posible crear depende del hardware.
- Cada hilo debe ser parte de un bloque y cada bloque de una grid.

	Compute Capability		
Parámetro	1.0, 1.1	1.2, 1.3	2.0
Número máximo de hilos por bloque	512	512	1024
Tamaño del *warp* (hilos)	32	32	32
Número máximo de *warps* residentes por MP	24	32	48
Número máximo de hilos residentes por MP	768	1024	1536
Registros de 32 bit por MP (Ki)	8	16	32
Memoria compartida por MP (KiB)	16	16	48
Bancos de memoria compartida	16	16	32
Memoria local por hilo (KiB)	16	16	512
Memoria constante (KiB)	64	64	64
Caché de memoria constante por MP (KiB)	8	8	8

PROGRAMACIÓN CUDA

- Para invocar a un Kernel se necesita establecer una configuración.
- El tamaño del grid y de los bloques , los cuales son determinados por el programador.



global void FuncionKernel( );
dim3 dimBlock(4,2,2); el número de threads por bloque
dim3 dimGrid(2,2,1); cantidad de bloques

FuncionKernel<<<dimGrid,dimBlock>>>

NOTA

Si se utiliza un grid unidimensional y los bloques son unidimensionales, no es necesario usar los dims.

Se puede poner directamente en la invocación.

__global__ void KernelFunc(...);

KernelFunc<<< 5000, 256 >>>(...);

Se lanza una ejecución con 5000 bloques y 256 hilos por bloque.

EJEMPLO 3 OTRO HOLA MUNDO

Bloque	hilos	
	0	a[0]=a[0]+b[0]
0	1	a[1]=a[1]+b[1]
	2	a[2]=a[2]+b[2]
	3	a[3]=a[3]+b[3]
	4	a[4]=a[4]+b[4]
	5	a[5]=a[5]+b[6]
	6	a[6]=a[7]+b[7]
	7	a[7]=a[8]+b[8]
	8	a[8]=a[9]+b[9]
	9	a[9]=a[10]+b[10]
	10	a[10]=a[11]+b[11]
	11	a[11]=a[12]+b[12]
	12	a[12]=a[13]+b[13]
	13	a[13]=a[14]+b[14]
	14	a[14]=a[15]+b[15]
	15	a[15]=a[16]+b[16]
	16	a[16]=a[17]+b[17]

El host imprimirá la palabra **Hola_**
y los núcleos formarán la palabra **mundo** para
Después escribir de forma completa
Hola_ Mundo

Configuración
1 bloque , 16 hilos por bloque

```
#include <stdio.h>
__global__ void kernel(){
}
void onDevice(){
      kernel <<<1,1>>> ( );
}
void vecAdd(float *h_A, float *h_B, float *h_C, int n){
//Colocar su código  aquí
}
void onHost(){
       int i,n=10;
       float *h_A;
       float *h_B;
       float *h_C;
//Colocar  código aquí
        for ( i=0;i<n; i++){
            h_A[i] = (float )i;
            h_B[i] =(float)(i+1);
        }
```

EJERCICIO
SÓLO EN EL HOST

```
 vecAdd(h_A,h_B,h_C,n);
for(i=0;i<n;i++){
   printf(" C[%d]= %lf ",i,h_C[i]);
}
     onDevice();
}
int main(){
     onHost();
     return 0;
}
```


EJEMPLO 4

El Kernel es ejecutado por un arreglo de hilos y todos lo hilos ejecutan el mismo código.

Si se requiere que cada hilo calcule un elemento de salida de la suma de vectores c[i]=a[i]+b[i], ¿Cuál es la expresión para el calcular el índice i utilizando los ids de hilos y bloques y blockDim?.

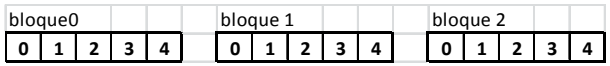
Si la configuración para el lanzamiento es

dim3 dimBlock(5,1,1); //número de threads por bloque

dim3 dimGrid(3,1,1); //cantidad de bloques

FuncionKernel<<<dimGrid,dimBlock>>>

Se asume un arreglo de 15 elementos



blockidx.x	threadidx.x	i
0	0	0
	1	1
	2	2
	3	3
	4	4
1	0	5
	1	6
	2	7
	3	8
	4	9
2	0	10
	1	11
	2	12
	3	13
	4	14

threadIdx.x=id hilo
blockIdx.x=id bloque
blockDim.x=hilos en el bloque

EJEMPLO 5

Se requiere que cada hilo calcule dos elementos adyacentes de la salida de la suma de vectores c[i]=a[i]+b[i] y c[i+1]=a[i+1]+b[i+1] , ¿Cuál es la expresión para calcular el índice i ?

Si la configuración para el lanzamiento es 5 hilos por bloque y 4 bloques en la grid.

Arreglo de 40 elementos

blockidx.x	threadidx.x	c[i]
0	0	c[0]
	0	c[0+1]
	1	c[2]
	1	c[2+1]
	2	c[4]
	2	c[5]
	3	c[6]
	3	c[7]
	4	c[8]
	4	c[9]
1	0	c[10]
	0	c[11]
	1	c[12]
	1	c[13]
	2	c[14]
	2	c[15]
	3	c[16]
	3	c[17]
	4	c[18]
	4	c[19]

blockidx.x	threadidx.x	c[i]
2	0	c[20]
	0	c[20+1]
	1	c[22]
	1	c[22+1]
	2	c[24]
	2	c[24+1]
	3	c[26]
	3	c[26+1]
	4	c[28]
	4	c[28+1]
3	0	c[30]
	0	c[30+1]
	1	c[32]
	1	c[32+1]
	2	c[34]
	2	c[34+1]
	3	c[36]
	3	c[37]
	4	c[38]
	4	c[39]

EJERCICIO

Se quiere que cada hilo calcule 2 elementos de la suma de vectores c[i]=a[i]+b[i].

Cada bloque debe procesar 2*blockDim.x elementos consecutivos (2*numbloques) que se dividen en dos secciones.

Cada bloque procesará la primera sección , donde cada hilo procesará un elemento, después el bloque procesará se la segunda sección, y cada hilo procesara otro elemento.

La variable i debe ser el índice al primer elemento a ser procesado (primera sección) por cada hilo en cada bloque.

¿Cual es la expresión para obtener i del primer elemento asignado a cada bloque?

Considerar 4 bloques en la grid y 4 hilos por bloque

blockidx.x	threadidx.x	sección 1	sección 2
0	0	c[0]	c[4]
	1	c[1]	c[5]
	2	c[2]	c[6]
	3	c[3]	c[7]
1	0	c[8]	c[12]
	1	c[9]	c[13]
	2	c[10]	c[14]
	3	c[11]	c[15]
2	0	c[16]	c[20]
	1	c[17]	c[21]
	2	c[18]	c[22]
	3	c[19]	c[23]
3	0	c[24]	c[28]
	1	c[25]	c[29]
	2	c[26]	c[30]
	3	c[27]	c[31]