CUDA编程2

1. 内置类型和函数 Built-ins and fucntions.
2. 线程同步 Synchronizing threads
3. 线程调度 Scheduling threads
4. 存储模型 Memory model
5. 重访 Matrix multiply
6. 原子函数 Atomic functions

## 函数的声明

\_\_global\_\_ void KernelFunc() device execute host call

\_\_device\_\_ float func()

\_\_host\_\_ float Hostfunc()

注：

Global 和 device 函数

1. 尽量少用递归
2. 不要用静态变量
3. 少用malloc
4. 小心通过指针实现的函数调用

## 向量数据类型

1. char[1-4], uchar[1-4]
2. short[1-4], ushort[1-4]
3. int[1-4], uint[1-4]
4. long[1-4], ulong[1-4]
5. longlong[1-4], ulonglong[1-4]
6. float
7. double1, double2

同时适用于host和device代码

通过函数 make\_<type name>构造

Int2 i2 = make\_int2(1, 2);

Float4 f4 = make\_float4(1.0f, 2.0f, 3.0f, 4.0f);

通过.x, .y, .z .w 访问

Int2 i2= make\_int2(1, 2);

Int x = i2.x;

Int y = i2.y;

## 数学函数

常见函数

Tool kit的函数手册中有提及

Intrinsic function 内建函数

1. 仅面向Device设备端
2. 更快、但精度更低
3. 以\_\_为前缀
4. \_\_exp, \_\_log \_\_sin, \_\_pow。。。。

线程层次

Int threadID = block.x \* blockDim.x + threadIdx.x; // 一维的线程

Float x = input[threadID];

Float y = func(x);

Output[threadID] = y;

## 线程同步

块内线程可以同步

调用\_\_syncthreads床架哪一个barrier栅栏

Mds[i] = Md[j];

\_\_syncthreads();

Func(Mds[i], Mds[i + 1]);

### Warp-块内的一组线程

一个warp的线程是天然同步的

Warp被要求天生一致

如果warp内部线程沿着不同分支执行，会有什么样的后果？

系统性能下降

G80/GT200-32线程

运行于同一个SM

线程调度的基本单位

ThreadIdx的值连续

一个实现细节-理论上

WarpSize

每个warp含32个线程， 但每个SM只有8个SPs。如何分配？

答：

当一个SM调度一个warp时，指令已经预备，在第一个周期8个线程进入SPs，在第二、第三、第四个周期各进入8个线程，因此， 分发一个warp需要4个线程。

## 内存模型

略

Local memory线程私有

Shared memory 一个block共有， 速度快

全局存储器Global Memory

1. 长延时
2. 片外、可读写
3. 随机访问影响性能
4. Host主机端可读写
5. GT200

带宽：150GB/s

容量：4GB

6、G80-86.4GB/s

常量存储器constant memory

短延时、高带宽， 当所有线程访问同一位置时只读

储存有global memory 但是有缓存

Host 主机端可读写

容量：64KB

储存的内容：比如pi的值，e的值

声明内存：

|  |  |  |  |
| --- | --- | --- | --- |
| 变量声明 | 存储器 | 作用域 | 生命期 |
| 必须是单独的变量而不能是数组（） | Register | Thread | Kernel |
| 自动变量数组 | Local | Thread | Kernel |
| \_\_shared\_\_ int shareVal | Shared | Thread | Kernel |
| \_\_device\_\_ int globalVal | Global | Grid | Application |
| \_\_constant\_\_int constantVar | Constant | Grid | application |

Global and constant 变量

Host可以通过以下函数访问：

CudaGetSymbolAddress（）

CudaGetSymbolSize（）

CudaMemcpyToSymbol()

cudaMemcpyFromSymbol()

constants变量必须在函数外声明