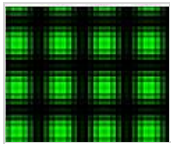
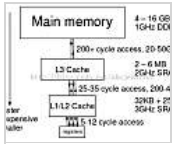
CUDA编程——Memory Coalescing

发表于: 2016-01-18 作者:junparadox 来源:转载 浏览:79

CUDA 编程 GPU

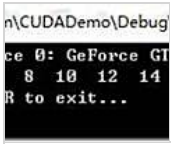

CUDA范例编程中的



CUDA系列学习(二


9. CUDA shared m


9. CUDA shared m



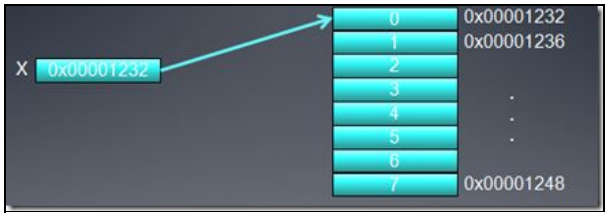
CUDA C编程探索

摘要: CUDA编程——MemoryCoalescing1GPU总线寻址介绍假定X是一个指向整数(32位整数)数组的指针,数组的首地址为0x00001232。一个线程要访问元素X[0],inttmp=X[0];假定memory总线宽度为256位,因为基于字节地址的总线要访问memeory,必须和总线宽度对齐,也就是说按必须32字节对齐来访问memory,比如访问0x00000000,0x00000020,

CUDA编程——Memory Coalescing

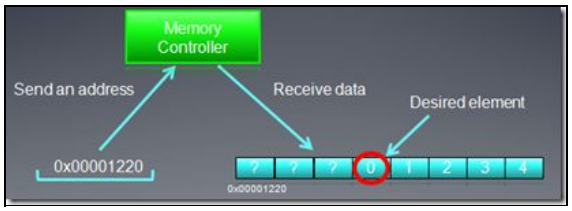
1 GPU总线寻址介绍

假定X是一个指向整数(32位整数)数组的指针,数组的首地址为0x00001232。一个线程要访问元素X[0],



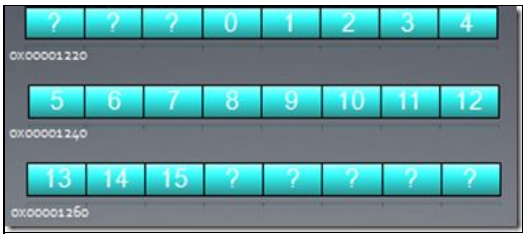
int tmp = X[0];

假定memory总线宽度为256位,因为基于字节地址的总线要访问memeory,必须和总线宽度对齐,也就是说按必须32字节对齐来访问memory,比如访问0x00000000,0x00000020,0x00000040,...等,所以我们要得到地址0x00001232中的数据,比如访问地址0x00001220,这时,它会同时得到0x00001220到 0x0000123F 的所有数据。只是对我们来说,只有一个32位整数有用,所以有用的数据是4个字节,其它28的字节的数据都被浪费了,白白消耗了带宽。



2 合并内存访问

为了利用总线带宽,GPU通常把多个线程的访存合并到较少的内存请求中去。假定多个线程访问32个字节以内的地址,它们的访问可以通过一个memory request完成,这样可以大大提高带宽利用率,在专业术语描述中这样的合并访问称作coalescing。



例如上面16个线程访问地址0x00001232 到 0x00001272, 我们只需要3次memory requst。

3 Bank Conflicts

对Nvidia GPUs来说, local memory是由banks组成的, 每个bank是32bit, 可视化图如下。bank是实际存储单元。每个bank在一次访存中, 可以被取址一次。并行访问相同的bank, 将导致访存串行(bank conflicts)。

```
Bank    |      1      |      2      |      3      |...
Address |  0  1  2  3 |  4  5  6  7 |  8  9 10 11 |...
Address | 64 65 66 67 | 68 69 70 71 | 72 73 74 75 |...
```

CUDA编程中,一个half-warp(16个threads)访问连续的32bit地址,不会有bank conflicts。一个例外情况是broadcast, 如果所有thread访问同一个地址, 内存只会被读一次, 并broadcast到所有threads。

CUDA编程——Memory Coalescing

0票 0票 0票 0票 0票 0票 0票 0票

开心 板砖 感动 有用 疑问 难过 无聊 震惊

0 0

顶 踩

10 CUDA C编程探索 摘要 App