CSC8014 Topics: GPU Programming
Student: Lina Mi

Homework #3

1. Explain the following different types of memories and their purposes:
    a. Shared memory
    b. Global memory
    c. Constant memory

Ans: a) Shared memory is memory allotted to each block, it works like cache in CPU, the data stored in the shared memory is visible to all threads within that block, this provides the means for the threads within one block could communicate and collaborate on computation between one another. Using qualifier __shared__ to make some variables reside in shared memory. Shared memory locates physically in GPU, and has high access speed, so it is effective as per-block, software-managed cache.

b)Global Memory is memory in GPU, much like RAM in CPU, using qualifier __global__ to declare variable or function means the variables resides in GPU or functions will run on GPU but can be called by CPU. The data stored in global memory is visible to all threads with application (include host). Global memory physically locates off chip of GPU.

c)Constant Memory is used to store the data that will not change over the course of a kernel execution and is read only. Using constant memory rather than global memory can reduce the required memory bandwidth.

2. For the attached CUDA code `dot.cu` for dot product, please answer the following questions:
    a. what is the purpose of the method __syncthreads() ?

    Ans: the method __syncthreads() is to guarantee that every threads that need to write to shared memory cache (storing temporary result in cache) complete the writing of cache before anyone tries to read these cache.

    b. In the kernel method `dot(…)`, the second __syncthreads() is executed for every thread even though only some of the threads are executing the instructions. Some smart people suggest the following updates (moving the __syncthreads() into if statement) to optimize the code

```
int i = blockDim.x/2;
while (i != 0) {
   if (cacheIndex < i) {
      cache[cacheIndex] += cache[cacheIndex + i];
      __syncthreads();
   }
   i /= 2;
}.
```

Do you think this change will work?

Ans: This change will not work, this change will cause GPU to stop responding and will force you to kill the program. Since only some threads (whose indexes satisfy the condition *cacheIndex<i*)could execute the codes inside the condition block *if(cacheIndex<i),* namely thread divergent, and *__syncthreads()* is used to guarantee that no thread could advance to an instruction beyond *__syncthreads()* until every thread in the block has executed *__syncthreads()*, when *__syncthreads()* was moved into this divergent branch, some of threads will never reach *__syncthreads()* and execute it. Therefore, because of guarantee that no instruction after *__syncthreads()* can be executed before every thread execute it, GPU will wait for those threads which can not reach *__syncthreads()* for ever.