LESSON 3

Atomics Operations and Histograms

TOPICS

- How to identify the use of atomic operations and their implementation.
- Ho to implement mutual exclusion (mutex)

Key words: atomic, histogram, atomicAdd(), atomicCAS(), atomicExch(), mutex.

Atomic Operations

- The atomic operations are based on the read-modify-write operations, for example the increment of a variable like x++.

The operation is atomic in the sense that it is guaranteed to be performed without interference from other threads.

- No other thread can access this address until the operation is complete.

- An atomic function performs a read-modify-write atomic operation residing in global or shared memory.

- CUDA C supports several atomic operations that allows to operate safely in memory, even when thousands of threads are competing for access
- Atomic operations on global memory are supported on GPUs of compute capability 1.1 or higher.
- Atomic operations on shared memory require a GPU of compute capability 1.2 or higher.
- It is necessary to inform to the compiler that the code will not run with a capability less than 1.1., in the case of atomics in global memory or 1.2, in the case of atomics in shared memory.

nvcc –arch=sm_11 or nvcc –arch=sm_12 -arch=sm_13

Limitations

- Only certain operations and data types are supported, mostly integer types.
- Not ordering in float operations

$$(A+B)+C \; != \; A+(B+C)$$

- The operations are kind of slow because of the serialize access to memory.

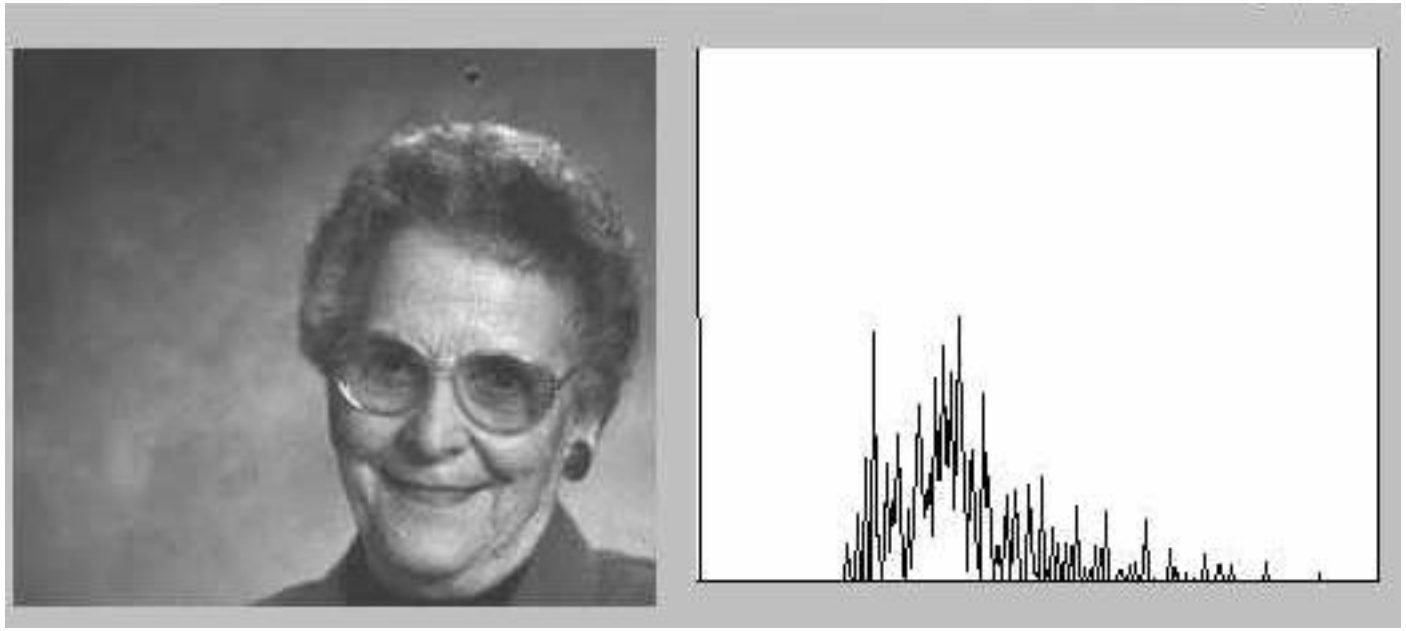
Example

- 00atomics: This example compares the result of a read-modify-write operation without atomics and with atomics.

Histograms

- A histogram is a representation of the distribution of data. It counts counts the number of observations that fall into each of the disjoint categories (known as *bins*).

- Basically, it shows the frequences or density in an interval of data.



Example

- 01histogram_gmem: This example shows the use of the atomic operation to generate a histogram.

Practice

- 02histogram_smem: This example must use share memory to optimize the histogram example.

Mutual Exclusion (mutex)

- It refers to the fact that two ore more threads or processes cannot access to a critical section at the same time.
- This technique could be used in the GPUs to protect an operation using atomic operations.
- It allows to perform arbitrary opertions on a memory location or data structure.

- The basic idea is to allocate a small piece of memory to be used as mutex.

    1. When a thread a 0 from the mutex , it interprets his value as a green light indicating that no other thread is using the memory.

    2. The thread marks with 1 (red light) the mutex, and it is free to lock the memory and make some operations without the intromission from the other threads.

- In CUDA, the function atomicCAS() is used to accomplish this operation correctly.

int atomicCas(int* address, int compare, int val)

where:

address: the function takes a pointer to memory
compare: a value with wich to compare the value at the location
val: a value to store in that location if the comparision is successful

Example

- 03reduction_mutex: This example makes use of the mutex technique in the dot operation.