Jack Melcher

67574625

EECS 117

HW3

1. Compile and run this code, which reports the input vector size, the time to execute the kernel, and an effective bandwidth. Record these data. Explain how the effective bandwidth is being calculated.

   Effective bandwidth is being calculated by taking the size of the array being reduced multiplied by the data type (float -> 4bytes) of the array and dividing it by time the GPU kernel took to execute multiplied by 1e9 to calculate an effective bandwidth in Gigabytes per second.

   Ex: (8388608*4)/(0.003353*1e9) = 10.01 GB/s

2. Implement this scheme in kernel1 of stride.cu. Measure and record the resulting performance.

   How much faster than the initial code is this version?

   Time to execute strided index GPU reduction kernel: 0.002211 secs

   Effective bandwidth: 15.18 GB/s

   The code is 1.5 times faster than the initial code

3. Implement this scheme in kernel2 of sequential.cu. Record the new effective bandwidth.

   Time to execute sequential index GPU reduction kernel: 0.001768 secs
   Effective bandwidth: 18.98 GB/s

4. Implement this scheme in kernel3 of first_add.cu and report the effective bandwidth.

   Time to execute first add GPU reduction kernel: 0.000983 secs
   Effective bandwidth: 34.13 GB/s

5. Implement this scheme in kernel4 of unroll.cu and report the effective bandwidth.

   Time to execute unrolled GPU reduction kernel: 0.000627 secs
   Effective bandwidth: 53.52 GB/s

6. Implement the algorithm cascading scheme in multiple.cu and report the effective bandwidth.

   Time to execute multiple add GPU reduction kernel: 0.000328 secs
   Effective bandwidth: 102.30 GB/s