Theodore Jagodits / CS 677 / HW #2 Report /

“I pledge my honor that I have abided by the Stevens Honor system”

1.4

a)

I start off with 256 threads, which equals to log2(256) = 8 thread synchronizations within my thread block. This is because I add the elements from global into the shared memory in the first computation, so I have only 256 elements to do the algorithm on, which halves every time until the stride >= 1. Every time it halves is when I must sync the threads.

b)

The minimum amount of operations needed is just 1. Since each thread loads from global and stores into shared, which would count as one operation. Following that, the maximum amount would be log2 of 512 which is 9. The thread 0 has to compute for every iteration which halves every time.

The average is (256 + 128\*2 + 64\*3 + 32\*4 + 16\*5 + 8\*6 + 4\*7 + 2\*8 + 9) /512 = 1.98 operations per thread. Each for loop of the operation loses half of threads that actually do work, so halve the number of threads and increment the number of operations every turn.

4.

a)

There is a total of 256 \* 1024 threads at the start. As there are vector n blocks of element n threads. = 262,144‬

b)

Each thread does 2 two loads and then do one store each. They access vectors A and B on line 21 and then store once on line 30.

c)

In each block, we access and store per thread accumResult, so that is already 256. Then we access in the for loop, which goes to zero and halves from 256/2 = 128. This brings the total of accesses

Math math math