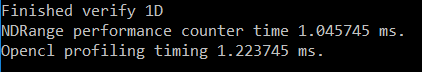
* Compare to SAXPY values obtained using Windows API performance counters. Explain difference

For 1D array

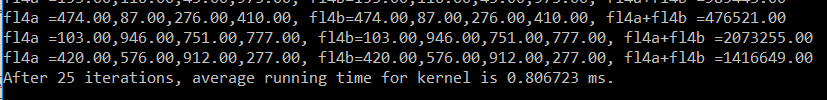


The opencl profiling timing is slower for 1D.



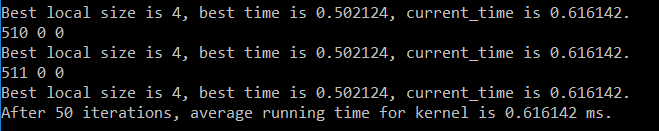
However the opencl profiling timing is faster in 2D. And it is because the opencl might have had optimized the 1D and the timing is off. On the 2D SAXPY, we can see that the Windows API performance counter counted for a longer time for the kernel. Because the windows API was measure the entire time to execute the kernel with a couple more if statements, while the OpenCL was measuring the enqueuer event only. Although they are different, they are similar.

* + 1. dot product of two float4 vectors
* kernel 1: manual dot product using vector component syntax

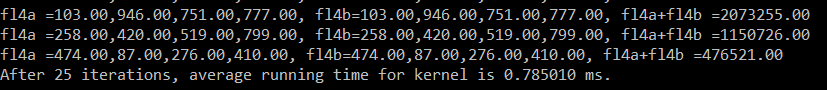




Best local size:

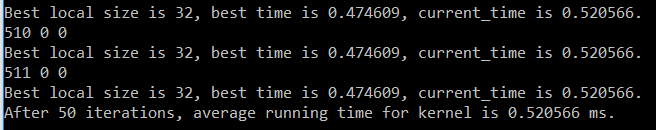


* kernel 2: use built-in geometric function [6.13.5]\* dot



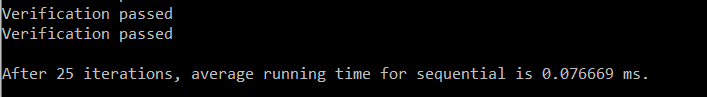


Best local size:

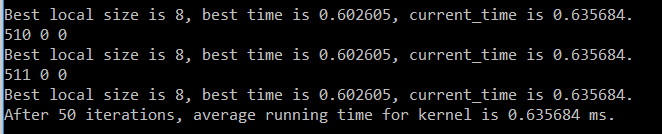


1. vector arithmetic A \* B + C

* FMA

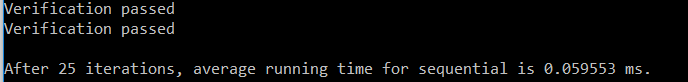


Best local size:

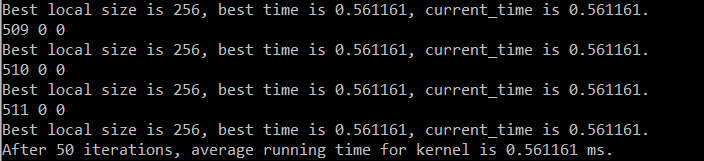


* mad



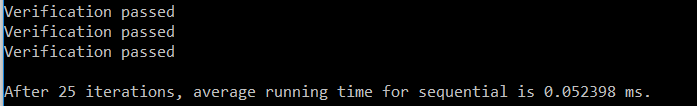


Best local size:

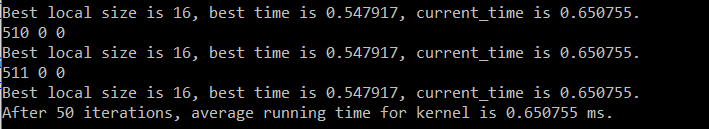


* manually





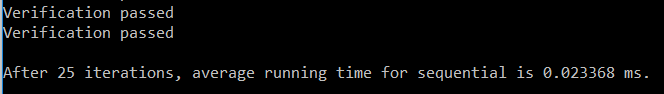
Best local size:



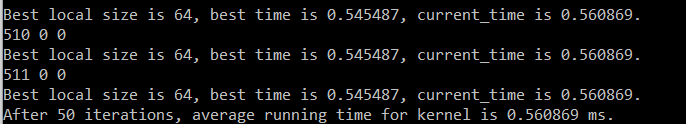
1. Cross product

* Build in float4 cross



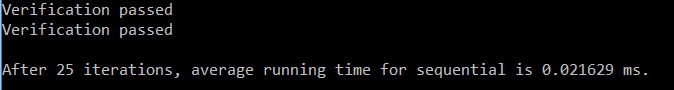


Best local size:

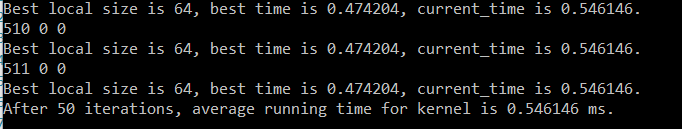


* .xyz syntax





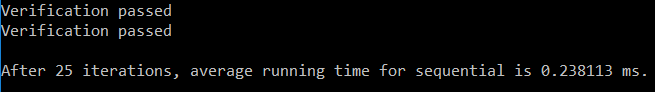
Best local size:



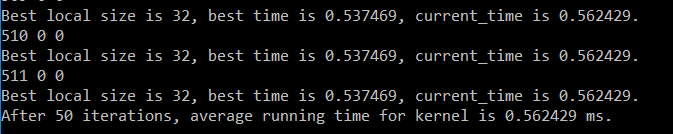
1. Square root

* Fast\_length()



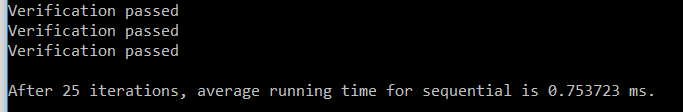


Best local size:

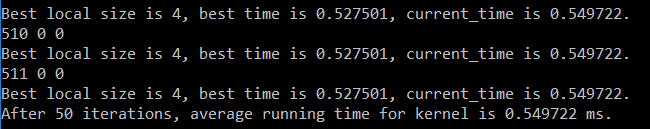


* Native\_sqrt()



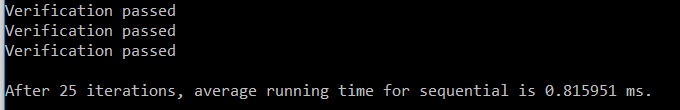


Best local size:



* Sqrt()





Best local size:

