

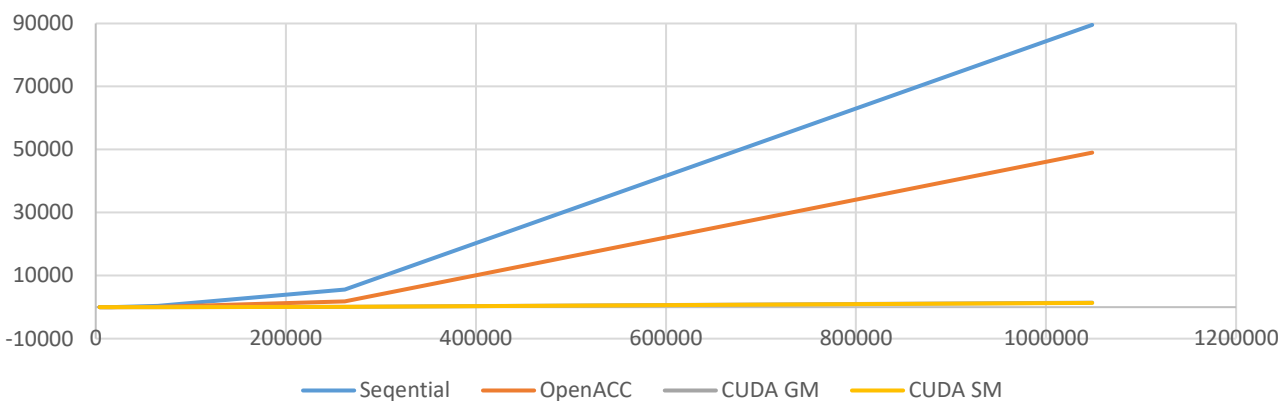
Size/System	Intel Core i3-7100 Dual Core @ 3.90GHz (CPU) Nvidia Geforce GTX 1050 TI (GPU)		
	Method	Speed(seconds)	Speedup(Ts/Tp)
<b>4,096</b>	Sequential	1.907368	
	OpenACC	0.454588	4.1958 (419.58 %)
	CUDA Global Memory	0.04443002	42.929 (4292.9 %)
	CUDA Shared Memory	0.04316807	44.184 (4418.6 %)
<b>16,384</b>	Sequential	24.24149	
	OpenACC	6.214528	3.9 (390 %)
	CUDA Global Memory	0.477581	50.758 (5075.8 %)
	CUDA Shared Memory	0.3654189	66.338 (6633.8 %)
<b>65,536</b>	Sequential	355.5284	
	OpenACC	98.96123	3.592 (359.2 %)
	CUDA Global Memory	5.696616	62.41 (6241 %)
	CUDA Shared Memory	5.450865	65.224 (6522.4 %)
<b>262,144</b>	Sequential	5617.434	
	OpenACC	1864.156	3.013 (301.3 %)
	CUDA Global Memory	88.55316	63.435 (6343.5 %)
	CUDA Shared Memory	83.92049	66.936 (6393.7 %)
<b>1,048,576</b>	Sequential	89563.81	
	OpenACC	49027.96	1.826 (182.6 %)
	CUDA Global Memory	1415.992	63.251 (6325.1 %)
	CUDA Shared Memory	1343.394	66.669 (6666.9 %)

There is a very significant speedup with the massively parallel approach (up to 66.9x) when using CUDA and shared memory in the GPU as recorded in the table above. The openACC approach is definitely faster than the sequential approach but not as fast as expected. The maximum speedup recorded using openACC was 4.19x, slowly decreasing as the problem size grows.

The openACC approach was no where near as fast as the optimized CUDA approach even when taking advantage of shared memory using OpenACC. This behavior was expected but not to that extent, as openACC was slower than optimized CUDA code when multiplying matrices but was at least 0.5x as fast CUDA Global Memory code, which is not the case when solving a radix-2 FFT due to the nature of the problem.

Additionally, using shared memory to solve the problem did not provide a significant performance boost. However, it did provide a minor boost (1.3x faster than the global memory code).

Radix 2 FFT Computation Time as problem size grows  
Linear Axis



Radix 2 FFT Computation Time as problem size grows  
Logarithmic Axis

