Project #6

OpenCL Array Multiply, Multiply-Add, and Multiply-Reduce

PangFa Chou | choupa@oregonstate.edu

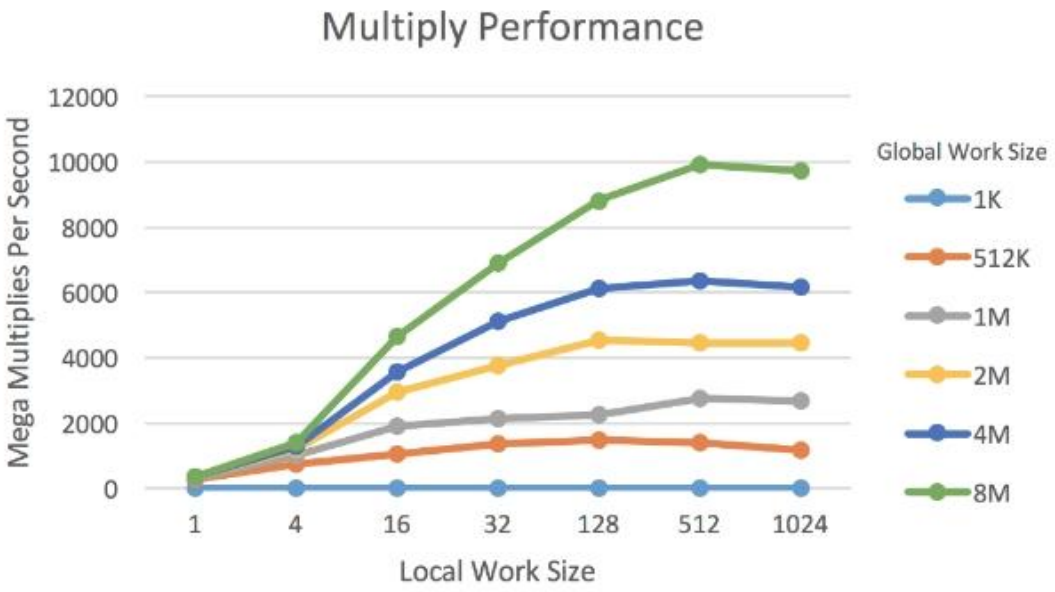
1. The program was run on a rabbit server.
2. The result of the program can be presented as following tables and graphs:

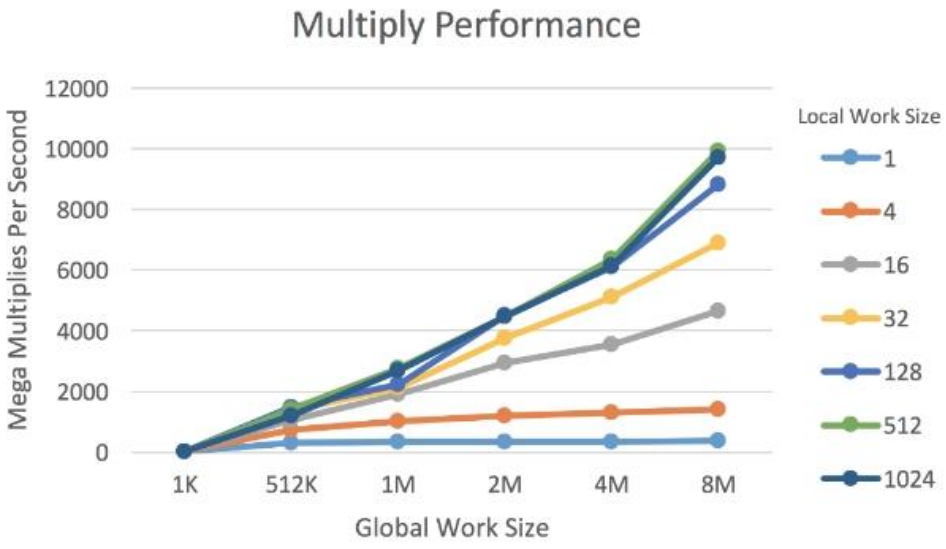
   Table:

Multiply

	A	B	C	D	E
1	1024	8	128	13.27	MegaMults/Sec
2	1024	16	64	17.35	MegaMults/Sec
3	1024	32	32	16.183	MegaMults/Sec
4	1024	64	16	15.439	MegaMults/Sec
5	1024	128	8	16.77	MegaMults/Sec
6	1024	256	4	15.468	MegaMults/Sec
7	1024	512	2	15.015	MegaMults/Sec
8	524288	8	65536	1310.075	MegaMults/Sec
9	524288	16	32768	611.12	MegaMults/Sec
10	524288	32	16384	1268.289	MegaMults/Sec
11	524288	64	8192	456.056	MegaMults/Sec
12	524288	128	4096	1606.846	MegaMults/Sec
13	524288	256	2048	1867.688	MegaMults/Sec
14	524288	512	1024	408.237	MegaMults/Sec
15	1048576	8	131072	1863.453	MegaMults/Sec
16	1048576	16	65536	2074.215	MegaMults/Sec
17	1048576	32	32768	2971.051	MegaMults/Sec
18	1048576	64	16384	404.937	MegaMults/Sec
19	1048576	128	8192	1763.599	MegaMults/Sec
20	1048576	256	4096	174.494	MegaMults/Sec
21	1048576	512	2048	2894.66	MegaMults/Sec
22	2097152	8	262144	311.772	MegaMults/Sec
23	2097152	16	131072	3791.537	MegaMults/Sec
24	2097152	32	65536	4011.485	MegaMults/Sec
25	2097152	64	32768	4726.541	MegaMults/Sec
26	2097152	128	16384	3899.038	MegaMults/Sec
27	2097152	256	8192	5918.64	MegaMults/Sec
28	2097152	512	4096	4324.453	MegaMults/Sec
29	4194304	8	524288	2405.211	MegaMults/Sec
30	4194304	16	262144	4247.108	MegaMults/Sec
31	4194304	32	131072	6196.297	MegaMults/Sec
32	4194304	64	65536	7770.669	MegaMults/Sec
33	4194304	128	32768	10311.293	MegaMults/Sec
34	4194304	256	16384	8482.269	MegaMults/Sec
35	4194304	512	8192	8504.87	MegaMults/Sec
36	8388608	8	1048576	2099.065	MegaMults/Sec
37	8388608	16	524288	4955.258	MegaMults/Sec
38	8388608	32	262144	6755.065	MegaMults/Sec
39	8388608	64	131072	7083.429	MegaMults/Sec
40	8388608	128	65536	12313.246	MegaMults/Sec
41	8388608	256	32768	11541.33	MegaMults/Sec
42	8388608	512	16384	12981.704	MegaMults/Sec
43					

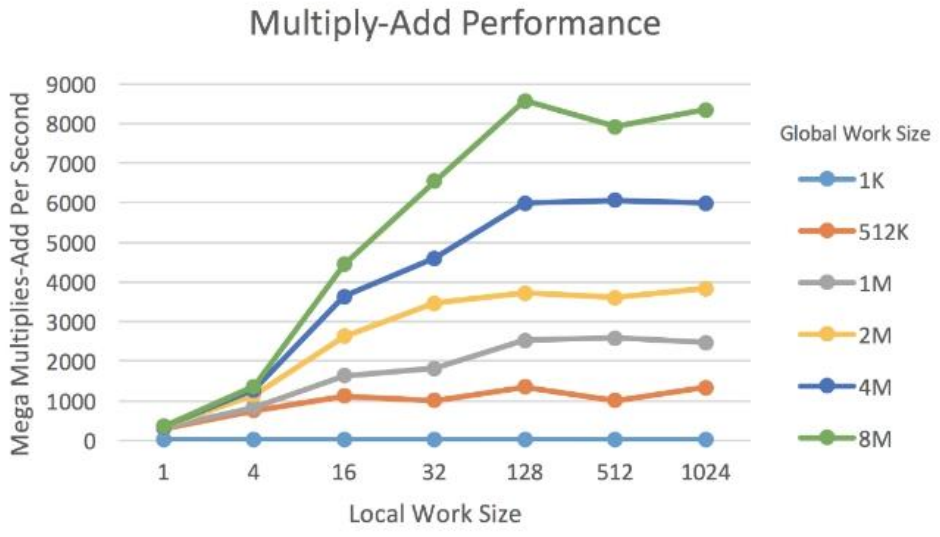
Graph:

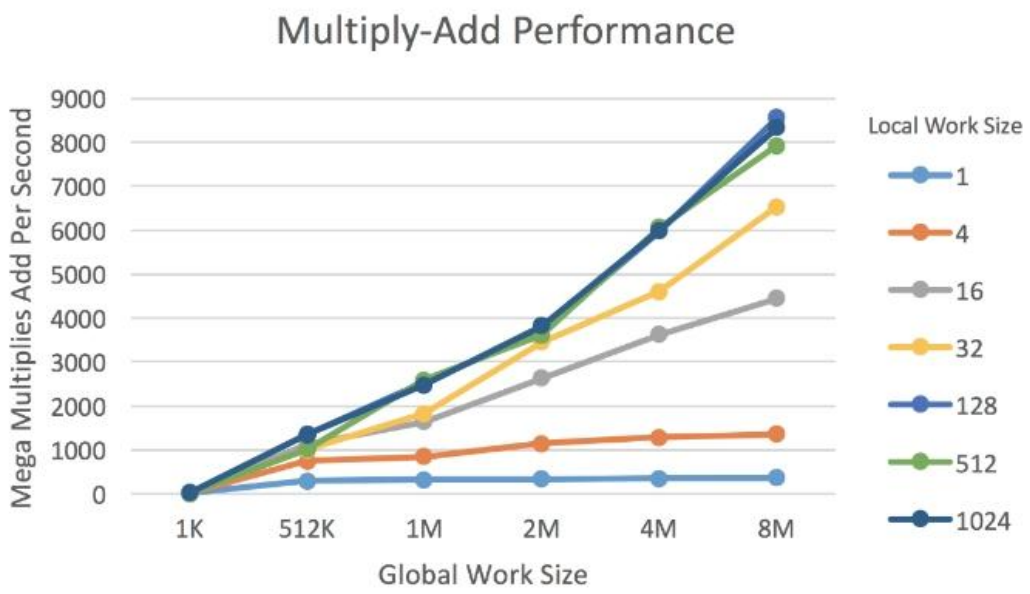




Multiply-Add

	A	B	C	D	E
1	1024	8	128	19.001	MegaMults/Sec
2	1024	16	64	14.117	MegaMults/Sec
3	1024	32	32	16.398	MegaMults/Sec
4	1024	64	16	22.022	MegaMults/Sec
5	1024	128	8	14.336	MegaMults/Sec
6	1024	256	4	15.938	MegaMults/Sec
7	1024	512	2	11.236	MegaMults/Sec
8	524288	8	65536	987.555	MegaMults/Sec
9	524288	16	32768	1699.965	MegaMults/Sec
10	524288	32	16384	1696.001	MegaMults/Sec
11	524288	64	8192	1705.678	MegaMults/Sec
12	524288	128	4096	1508.13	MegaMults/Sec
13	524288	256	2048	1756.161	MegaMults/Sec
14	524288	512	1024	1364.75	MegaMults/Sec
15	1048576	8	131072	1566.421	MegaMults/Sec
16	1048576	16	65536	2557.427	MegaMults/Sec
17	1048576	32	32768	3335.208	MegaMults/Sec
18	1048576	64	16384	2786.906	MegaMults/Sec
19	1048576	128	8192	3669.789	MegaMults/Sec
20	1048576	256	4096	3274.528	MegaMults/Sec
21	1048576	512	2048	3772.372	MegaMults/Sec
22	2097152	8	262144	2179.944	MegaMults/Sec
23	2097152	16	131072	3154.656	MegaMults/Sec
24	2097152	32	65536	4627.717	MegaMults/Sec
25	2097152	64	32768	4929.407	MegaMults/Sec
26	2097152	128	16384	4759.591	MegaMults/Sec
27	2097152	256	8192	4458.478	MegaMults/Sec
28	2097152	512	4096	4176.936	MegaMults/Sec
29	4194304	8	524288	2276.094	MegaMults/Sec
30	4194304	16	262144	4142.273	MegaMults/Sec
31	4194304	32	131072	5600.455	MegaMults/Sec
32	4194304	64	65536	7010.393	MegaMults/Sec
33	4194304	128	32768	5316.682	MegaMults/Sec
34	4194304	256	16384	8157.252	MegaMults/Sec
35	4194304	512	8192	7355.447	MegaMults/Sec
36	8388608	8	1048576	2620.09	MegaMults/Sec
37	8388608	16	524288	4704.215	MegaMults/Sec
38	8388608	32	262144	7534.565	MegaMults/Sec
39	8388608	64	131072	8882.577	MegaMults/Sec
40	8388608	128	65536	10108.011	MegaMults/Sec
41	8388608	256	32768	10438.045	MegaMults/Sec
42	8388608	512	16384	1384.314	MegaMults/Sec





3. OvFor global workloads, performance improves when local workloads increase when the local job size equals 128, and the job size increases and performance peaks. For a given local workload, performance increases with the global workload.
4. There are many processing elements idle and lots of compute time is wasted when the local work size is too small.
5. They basically have the same performance, but the Multiply is slightly better. I think it is because Multiply-Add is more complicated than Multiply, so the processing time of doing Multiply-Add is longer.
6. Before starting work, it is necessary to check work items' sizes because dif- ferent work sizes impact the different performances. If the data size is too small, it is unworthy to do it on GPU.

Part2

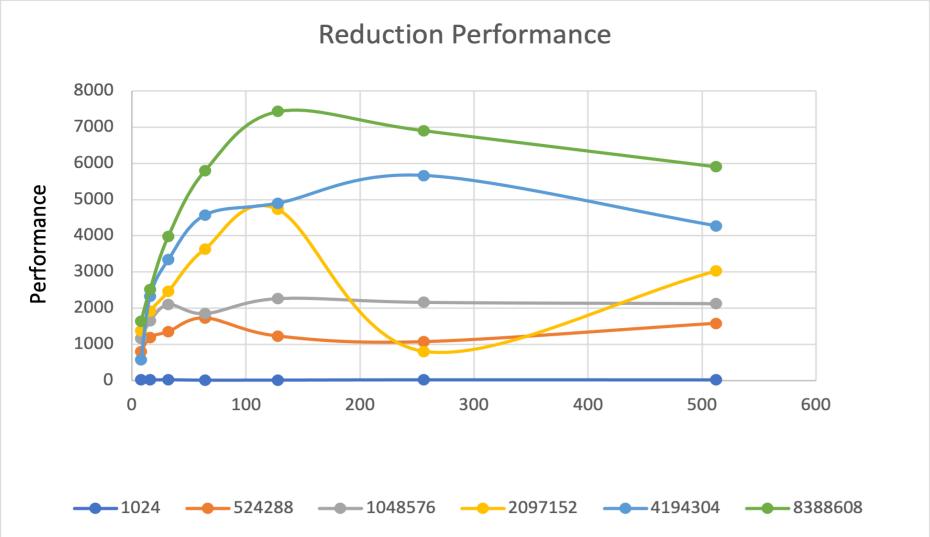
7.

Multiply Reduction

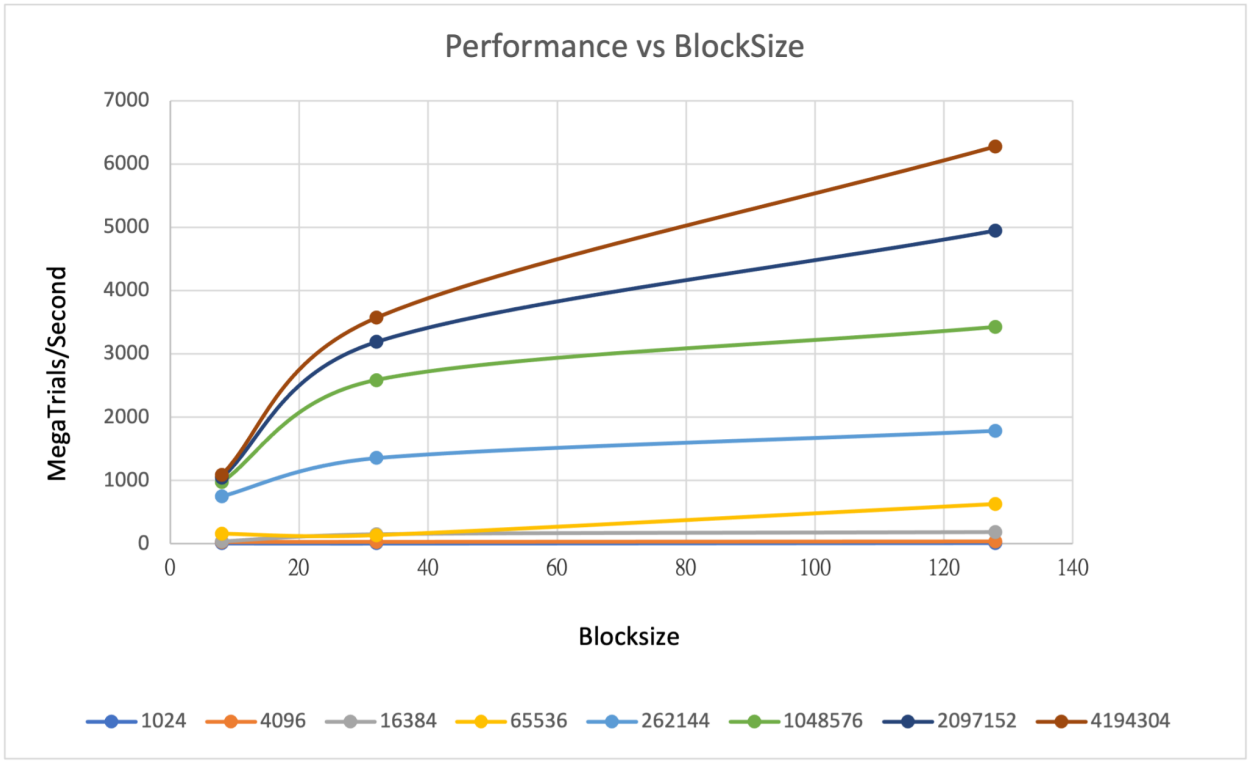
Table:

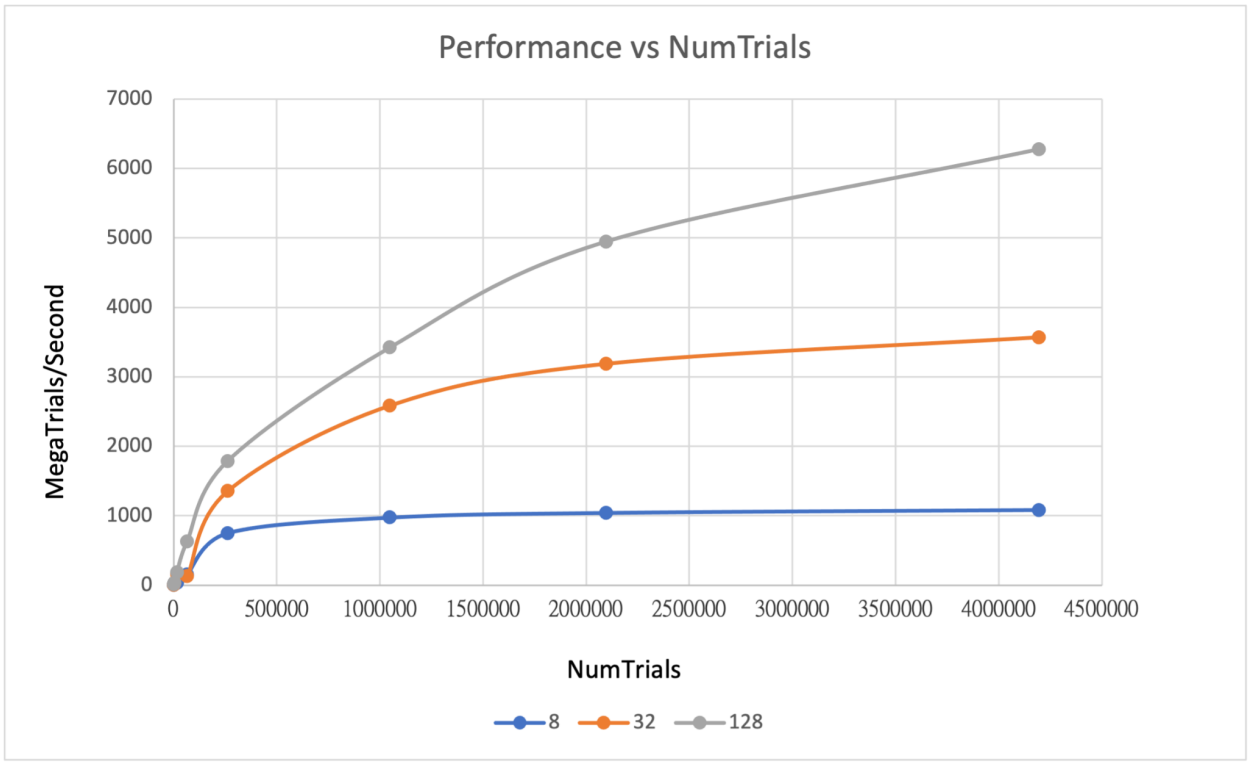
	A	B	C	D	E
1	1024	8	128	14.228	MegaMults/Sec
2	1024	16	64	13.718	MegaMults/Sec
3	1024	32	32	18.671	MegaMults/Sec
4	1024	64	16	10.244	MegaMults/Sec
5	1024	128	8	11.3	MegaMults/Sec
6	1024	256	4	17.256	MegaMults/Sec
7	1024	512	2	15.628	MegaMults/Sec
8	524288	8	65536	796.829	MegaMults/Sec
9	524288	16	32768	1190.26	MegaMults/Sec
10	524288	32	16384	1349.863	MegaMults/Sec
11	524288	64	8192	1723.526	MegaMults/Sec
12	524288	128	4096	1226.06	MegaMults/Sec
13	524288	256	2048	1071.581	MegaMults/Sec
14	524288	512	1024	1576.55	MegaMults/Sec
15	1048576	8	131072	1164.404	MegaMults/Sec
16	1048576	16	65536	1644.147	MegaMults/Sec
17	1048576	32	32768	2097.013	MegaMults/Sec
18	1048576	64	16384	1849.67	MegaMults/Sec
19	1048576	128	8192	2263.184	MegaMults/Sec
20	1048576	256	4096	2160.533	MegaMults/Sec
21	1048576	512	2048	2124.352	MegaMults/Sec
22	2097152	8	262144	1377.181	MegaMults/Sec
23	2097152	16	131072	1908.052	MegaMults/Sec
24	2097152	32	65536	2458.837	MegaMults/Sec
25	2097152	64	32768	3634.277	MegaMults/Sec
26	2097152	128	16384	4729.483	MegaMults/Sec
27	2097152	256	8192	798.872	MegaMults/Sec
28	2097152	512	4096	3027.106	MegaMults/Sec
29	4194304	8	524288	565.738	MegaMults/Sec
30	4194304	16	262144	2325.078	MegaMults/Sec
31	4194304	32	131072	3335.749	MegaMults/Sec
32	4194304	64	65536	4567.421	MegaMults/Sec
33	4194304	128	32768	4898.16	MegaMults/Sec
34	4194304	256	16384	5665.039	MegaMults/Sec
35	4194304	512	8192	4272.904	MegaMults/Sec
36	8388608	8	1048576	1631.149	MegaMults/Sec
37	8388608	16	524288	2518.819	MegaMults/Sec
38	8388608	32	262144	3985.272	MegaMults/Sec
39	8388608	64	131072	5792.485	MegaMults/Sec
40	8388608	128	65536	7433.704	MegaMults/Sec
41	8388608	256	32768	6905.682	MegaMults/Sec
42	8388608	512	16384	5906.784	MegaMults/Sec

Graphs:



8. Overall, every line is stable. However, when the line comes to 128, the performance becomes worse.
9. The reason might be when size 128. The GPU is not so busy and the overhead may cost too much time
10. It is necessary to check the size before starting work because even though it just changes a little bit, it is possible to have big change. If the data size is too small, it is not worth to do it on GPU.


Performance vs BlockSize
MegaTrials/Second
7000
6000
5000
4000
3000
2000
1000
0
0
20
40
60
80
100
120
140
Blocksize
1024
4096
16384
65536
262144
1048576
2097152
4194304


Performance vs NumTrials
MegaTrials/Second
7000
6000
5000
4000
3000
2000
1000
0
0
500000
1000000
1500000
2000000
2500000
3000000
3500000
4000000
4500000
NumTrials
8
32
128

11. We can see that the performance grows up really quick at first. Then, after they get bigger, the performance gradually approaches a certain value.
12. I think the reason why performance grows up so fast is because the more block size it has, its parallelism is better. Then, when the numtrials is larger, it starts to hit the peak performance so they gradually approach certain values.
13. It is because its block size is smaller. It exposes more parallelism when the blocksize is higher.
14. There is a difference between Project 1 and Project 5. The performance in project 1 grows up and drops at the beginning, and then grows up again. It is because of the false sharing problem, which the higher number of threads may cause worse performance at some point.
15. We should assign more block size in GPU parallel computing because it performs better.