Page: Details | Result: 2 - 127 - addRowSectionsKernel | Add Baseline | Apply Rules | Occupancy Calculator | Save as PDF

	Result	Time	Cycles	Regs	GPU	SM Frequency	CC	Process
Current	127 - addRowSectionsKernel (6, 4001, 1)x(1024, 1, 1)	538.34 usecond	٧٤٢٬٨٢٦	١٦	0 - NVIDIA GeForce RTX 3090	1.38 cycle/nsecond	8.6	[258893] task2

GPU Speed Of Light Throughput

High-level overview of the throughput for compute and memory resources of the GPU. For each unit, the throughput reports the achieved percentage of utilization with respect to the theoretical maximum. Breakdowns show the throughput for each individual sub-metric of Compute and Memory to clearly identify the highest contributor.

Compute (SM) Throughput [%]	23.96	Duration [usecond]	538.34
Memory Throughput [%]	78.41	Elapsed Cycles [cycle]	742,826
L1/TEX Cache Throughput [%]	22.43	SM Active Cycles [cycle]	693,517.38
L2 Cache Throughput [%]	34.43	SM Frequency [cycle/nsecond]	1.38
DRAM Throughput [%]	78.41	DRAM Frequency [cycle/nsecond]	9.41

High Memory Throughput Memory is more heavily utilized than Compute: Look at the Memory Workload Analysis section to identify the DRAM bottleneck. Check memory replay (coalescing) metrics to make sure you're efficiently utilizing the bytes transferred. Also consider whether it is possible to do more work per memory access (kernel fusion) or whether there are values you can (re)compute.

Launch Statistics

Summary of the configuration used to launch the kernel. The launch configuration defines the size of the kernel grid, the division of the grid into blocks, and the GPU resources needed to execute the kernel. Choosing an efficient launch configuration maximizes device utilization.

Grid Size	24,006	Function Cache Configuration	cudaFuncCachePreferNone
Registers Per Thread [register/thread]	16	Static Shared Memory Per Block [byte/block]	0
Block Size	1,024	Dynamic Shared Memory Per Block [byte/block]	0
Threads [thread]	24,582,144	Driver Shared Memory Per Block [Kbyte/block]	1.02
Waves Per SM	292.76	Shared Memory Configuration Size [Kbyte]	8.19

Occupancy

Occupancy is the ratio of the number of active warps per multiprocessor to the maximum number of possible active warps. Another way to view occupancy is the percentage of the hardware's ability to process warps that is actively in use. Higher occupancy does not always result in higher performance, however, low occupancy always reduces the ability to hide latencies, resulting in overall performance degradation. Large discrepancies between the theoretical and the achieved occupancy during execution typically indicates highly imbalanced workloads.

Theoretical Occupancy [%]	66.67	Block Limit Registers [block]	4
Theoretical Active Warps per SM [warp]	32	Block Limit Shared Mem [block]	8
Achieved Occupancy [%]	56.30	Block Limit Warps [block]	1
Achieved Active Warps Per SM [warp]	27.02	Block Limit SM [block]	16

Occupancy Limiters This kernel's theoretical occupancy (66.7%) is limited by the number of warps within each block The difference between calculated theoretical (66.7%) and measured achieved occupancy (56.3%) can be the result of warp scheduling overheads or workload imbalances during the kernel execution. Load imbalances can occur between warps within a block as well as across blocks of the same kernel. See the CUDA Best Practices Guide for more details on optimizing occupancy.