Result	Date	Time	Target	Platform	Configuration	Duration
Pass	Today, 10:24 PM	1:32:24.6	Tutorial_Matrix_Multiply_Step_1	Intel(R) Arc(TM) Graphics	Release	2:19:04.2

Summary | Details | Source | Control | Comments | Raw | Systems

Compare | Hide | Filter | Report

GPU Speed-of-Light Throughput

Top utilized pipelines or hardware units. Memory throughput and compute throughput are the two high-level indicators of how well a kernel is using the GPU's memory and compute resources. Memory throughput reflects how efficiently data is being read from and written to the GPU's memory subsystem. Compute throughput reflects how efficiently the GPU's compute units are used for arithmetic and logic operations. Low values in either suggest the kernel is not saturating the hardware and may be limited by other factors, such as latency or synchronization.

Compute (SM) Throughput [%]	1.21
Memory [%]	73.22
L1/TEX Cache Throughput [%]	73.22
L2 Cache Throughput [%]	17.84
DRAM Throughput [%]	1.21
Latency Issue	this kernel exhibits low compute throughput and memory bandwidth utilization relative to the peak performance of this device. Achieved compute throughput and/or memory bandwidth below 60.0% of peak typically indicate latency issues. Look at scheduler statistics and warp state statistics for potential reasons.

Roofline Analysis The ratio of peak float (fp32) to double (fp64) performance on this device is 32:1. The kernel achieved 0% of this device's fp32 peak performance and 0% of its fp64 peak performance. See the Roofline Analysis chart for more details on roofline analysis.

GPU Throughput

Speed Of Light [%]

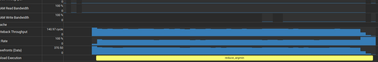
Compute Throughput Breakdown

Int Inst Executed Per Cycle [%]	1.47
SM Issue Active [%]	1.21
Int Inst Executed Per Active Cycle [%]	1.47
Int Pipe Fully Utilized Cycles Active [%]	0.00
SM Pipe Fully Utilized Cycles Active [%]	0.00
SM Inst Issued [%]	1.21
SM Pipe Inst Cycles Active [%]	1.21
Int Inst Executed Per Issue Slot [%]	1.47
Int Inst Executed Per Cycle Peak [%]	0.75
Int Inst Executed Per Cycle [%]	1.47
Int Inst Executed Per Issue Slot Peak [%]	0.75
Int Inst Executed Per Issue Slot [%]	1.47
Int Pipe Active Cycles Peak Activity [%]	0.01
Int Pipe Active Cycles [%]	1.21
Int Pipe Shared Cycles Active [%]	0.00
Int Pipe Fp64 Cycles Active [%]	0.00
FC Request Cycles Active [%]	0.00
Int Pipe Cycles Lost Due To Shared Activity [%]	0.00
Int Pipe Cycles Lost Due To Pipe Activity [%]	0.00
Int Inst Executed Per Cycle Peak [%]	0.75
Int Inst Executed Per Issue Slot Peak [%]	0.75

Memory Throughput Breakdown

L1/TEX Hit Rate [%]	0.00
L1 Sectors Misses To Global Per Cycle [%]	0.00
L1 Wavefronts Shared Per Cycle [%]	0.00
L1 Sectors Misses To Global Per Wavefront [%]	2.00
L1 Wavefronts Shared Per Wavefront [%]	2.00
L1 Sectors Misses To Global [%]	0.00
L1 Wavefronts Shared [%]	0.00
L1/TEX Cache Throughput [%]	73.22
L1 Sectors Misses To Global Per Cycle Peak [%]	0.00
L1 Wavefronts Shared Per Cycle Peak [%]	0.00
L1 Sectors Misses To Global Per Wavefront Peak [%]	2.00
L1 Wavefronts Shared Per Wavefront Peak [%]	2.00
L1 Sectors Misses To Global Peak [%]	0.00
L1 Wavefronts Shared Peak [%]	0.00
L1/TEX Cache Throughput Peak [%]	73.22
GPU Compute Memory Workload Throughput Internal Active [%]	1.21

Floating Point Operations Roofline



FM Sampling

Provides a view of FM samples aggregated periodically over the workload duration. Data is collected across multiple passes. Use this section to understand how workload behavior changes over its runtime.

Maximum Sampling Interval [cycle]	20,000	# Pass Groups	1
Maximum Buffer Size [Megabyte]	0.95	# Dropped Samples [sample]	0

Compute Workload Analysis

Detailed analysis of the compute resources of the streaming multiprocessor (SM), including the achieved instructions per clock (IPC) and the utilization of each available pipeline. Pipelines with very high utilization might limit the overall performance.

Executed Ipc Elapsed [inst/cycle]	0.08
Executed Ipc Active [inst/cycle]	1.47
Issued Ipc Active [inst/cycle]	1.47

Low Utilization All compute pipelines are under-utilized. Either the kernel is very small or it doesn't issue enough warps per scheduler. Check the Launch Statistics and Scheduler Statistics sections for further details.

Memory Workload Analysis

Detailed analysis of the memory resources of the GPU. Memory can become a limiting factor for the overall kernel performance when fully utilizing the control hardware units (Mem Busy), exhausting the available communication bandwidth between those units (Max Bandwidth), or by reaching the maximum capacity of the memory hardware units (Mem Pipes Busy). Example: If all of the memory units (Mem Pipes Busy) are busy, but only a small fraction of the maximum bandwidth (Max Bandwidth) is used, the kernel is likely bottlenecked by the memory pipelines.

Memory Throughput [%]	73.22	Mem Busy [%]	1.21
L1/TEX Hit Rate [%]	0.00	Max Bandwidth [%]	3.47
Lts Hit Rate [%]	0.00	Mem Pipes Busy [%]	1.21

Memory L2 Compression The optimal metric lts_sectors_compressed_ratio_sum is not available for this kernel. Collecting it as an additional metric could enable the rule to provide more guidance.

Shared Load Bank Conflicts The ratio of shared load bank conflicts to shared load wavefronts is 2. This indicates that on average, every 1 out of every 2 L1 Cache transactions per sector are utilized by each thread. This could possibly be caused by a strided shared memory access. Check the Source Counters section for uncoalesced shared accesses.

L1/TEX Global Store Access Pattern The memory access pattern for global stores to L1/TEX might not be optimal. On average, only 8.0 of the 32 bytes transmitted per sector are utilized by each thread. This could possibly be caused by a strided memory access. Check the Source Counters section for uncoalesced global stores.

Shared Load Bank Conflicts The memory access pattern for shared loads might not be optimal and causes on average a 1.0 - way bank conflict across all 14724 shared load requests. This results in 14219 bank conflicts, which represent 12.70% of the overall 111950 wavefronts for shared loads. Check the Shared Memory section for uncoalesced shared loads.

Shared Memory

	Instructions	Requests	Wavefronts	% Peak	Bank Conflicts
Shared Load	46,904	94,904	111,950	0.38	14,219
Shared Load Matrix	0	0	0		
Shared Store	102,992	102,992	155,206	0.52	0
Shared Atomic	0	0	0		
Shared Atomic Compare Swap	0	0	136,000	0.46	0
Total	157,456	197,456	404,556	1.36	14,219

L1/TEX Cache

	Instructions	Requests	Wavefronts	% Peak	Sectors	Sectors/Req	Hit Rate	Bytes	Sector Misses to L2	% Peak to L2	Returns to SM
Global Load	63,700	63,700	63,700	0.21	1,020,200	16.01	0.00	4,227,072	153,200	0.45	153,984
Surface Load	0	0	0		0			0	0		
Texture Load	0	0	0		0			0	0		
Global Store	1,024	1,024	1,024	0.00	8,192	8.00	0.00	16,896	1,024	0.00	
Global Atomic	0	0	0		0			0			
Surface Store	0	0	0		0			0			
Global Reduction	0	0	0		0			0			
Surface Atomic	0	0	0		0			0			
Global Atomic CAS	0	0	0		0			0			
Surface Atomic CAS	0	0	0		0			0			
Total	65,724	65,724	65,724	0.21	1,028,392	15.64	0.00	4,227,072	154,224	0.45	153,984
Pipe Lsu Mem Global Op Ld	63,700	63,700	63,700	0.21	1,020,200	16.01	0.00	4,227,072	153,200	0.45	
Pipe Lsu Mem Global Op St	1,024	1,024	1,024	0.00	8,192	8.00	0.00	16,896	1,024	0.00	
Uncached Global Load	0	0	0		0			0	0		see above
Global Store Bypass	0	0	0		0			0	0		see above
Global Atomic Bypass	0	0	0		0			0	0		see above
Texture Load Bypass	0	0	0		0			0	0		
Surface Load Bypass	0	0	0		0			0	0		
Total Bypass	0	0	0		0			0	0		
Global Hit	65,724	65,724	65,724	0.21	1,028,392	15.64	0.00	4,227,072	153,224	0.45	153,984
Global Miss	0	0	0		0			0	0		
Texture Hit	0	0	0		0			0	0		
Texture Miss	0	0	0		0			0	0		
Total Hit	65,724	65,724	65,724	0.21	1,028,392	15.64	0.00	4,227,072	153,224	0.45	153,984
Total Miss	0	0	0		0			0	0		
Pipe Tex Mem Texture Op Tex	0	0	0		0			0	0		
Pipe Tex Mem Texture Op Ld	0	0	0		0			0	0		
Pipe Tex Mem Texture Op St	0	0	0		0			0	0		
Pipe Tex Mem Texture Op Cx	0	0	0		0			0	0		
Pipe Tex Mem Texture Op Cl	0	0	0		0			0	0		
Pipe Tex Mem Texture Op Gr	0	0	0		0			0	0		
Pipe Tex Mem Texture Op Gh	0	0	0		0			0	0		

L2 Cache

	Requests	Sectors	Sectors/Req	% Peak	Hit Rate	Bytes	Throughput	Sector Misses to Device	Sector Misses to System	Sector Writes to Peer
L1/TEX Load	65,724	153,224	2.33	0.45	0.17	4,227,072	1,186,443.05 B/s	127,272	0	0
L1/TEX Atomic ALU	1,024	1,024	1	0.00	99.01	16,928	28,503,209.28	0	0	0
L1/TEX Atomic CAS	48	144	3	0.00	0.00	4,608	7,756,362.24 B/s	144	0	0
L1/TEX Atomic CAS	96,000	96,138	1.00	0.28	100.00	2,996,416	5,044,010,111.68	0	0	0
L1/TEX Reduction	0	0	0			0	0 B/s	0	0	0
Total	162,772	250,530	1.54	0.73	58.92	7,244,624	5,163,712,686.25 B/s	127,416	0	0
EQ Total	153,504	244,504	1.59	0.68	58.92	7,244,624	5,163,712,686.25 B/s	127,416	0	0
GPU Total	228,752	250,530	1.10	0.68	58.92	7,244,624	5,163,712,686.25 B/s	127,416	0	0

Device Memory

	Sectors	% Peak	Bytes	Throughput
Load	165,640	1.10	5,300,608	4,534,247.14 B/s
Store	24,000	0.16	770,816	659,174.64 B/s
Total	189,640	1.26	6,071,424	5,193,421.78 B/s

Scheduler Statistics

Summary of the activity of the schedulers issuing instructions. Each scheduler maintains a pool of warps that it can issue instructions for. The upper level of warps in the pool (Theoretical Warps) is limited by the launch configuration. On every cycle each scheduler checks the state of all of its warps in the pool (Eligible Warps), and chooses one of those to issue a single instruction. Fewer Eligible Warps indicate that none of the warps in the pool can issue an instruction. Issued Warps, On cycles with no eligible warps, the issue slot is skipped and no instruction is issued. Having many skipped issue slots indicates poor latency hiding.

One Or More Eligible	99.90
Issue Active	1.21
Issued Warps Per Active	1.21

Issue Slot Utilization Every scheduler is capable of issuing one instruction per cycle, but for this kernel each scheduler only issues an instruction every 1.2 cycles. This might have bottlenecks in other parts of the kernel instead causing warps to be not ready to issue an instruction. Issuing more than one instruction per cycle, if supported by the kernel's instruction stream, might be a viable optimization to increase the number of eligible warps, where the ratio of active warps to eligible to issuing to issued is closer to 1:1.

Issue Slot Utilization	1.21
Eligible Warps Per Scheduler	1.21
Issued Warps Per Scheduler	1.21

Warps Per Scheduler

Warp State Statistics

Analysis of the stalls in which all warps spent cycles during the kernel execution. The warps states describe a warp's readiness or inability to issue its next instruction. The warp cycles per instruction defines the latency between two consecutive instructions. The higher the value, the more cycles on average all warps had to wait to issue their next instruction. For each cycle when the kernel is waiting for at least one warp, the reason for each warp preventing the warp from being eligible to issue an instruction is recorded. The more warps are waiting for a given stall reason, the more this stall reason may be the bottleneck for kernel performance. Some instructions first need to wait for other instructions to be executed before they can be issued. Stalls caused by long dependencies can indicate that memory access or math pipelines are the bottleneck.

Warp Cycles Per Issued Instruction [cycle]	19.91
Warp Cycles Per Executed Instruction [cycle]	19.98
Avg. Active Threads Per Warp	32
Avg. Not Predicated Off Threads Per Warp	29.98

Long Scoreboard Stalls On average, each warp of this kernel spends 9.0 cycles being stalled waiting for a scoreboard dependency on a L1TEX (local, global, surface, texture) operation. Find the instruction producing the data being waited upon to identify the cause. To reduce these stalls, cache the accessed data in smaller chunks or reorder memory accesses to improve data locality. Consider using asynchronous copies, e.g. through LDGSTS, LDGDEPBAR/DEPBAR or the cuda::memcpy_async() API, to bypass the L1 cache. **Wait** stalls can be reduced by specifying a smaller number of outstanding memory instructions. Consider reducing frequently used data to shared memory. This stall type represents about 45.4% of the total average of 19.9 cycles between issuing two instructions.

Warp Stall Check the Warp Stall Sampling (All Cycles) table for the top stall locations in your source based on sampling data. The Warp Stalls section provides more details on each stall reason.

Thread Divergence Instructions are executed in warps, which are groups of 32 threads. Optimal instruction throughput is achieved if all 32 threads of a warp execute the same instruction. The chosen launch configuration, early thread completion, and divergent flow control can significantly lower the number of active threads in a warp per cycle. This kernel achieves an average of 29.9 threads being active per cycle. This is further reduced to 29.7 threads per warp due to predication. The compiler may use predication to avoid an actual branch. Instead, all instructions are scheduled, but a per-thread condition code or predicate controls which threads execute the instructions. Try to avoid different execution paths within a warp where possible.

Avg. Threads Executed	29.9
Avg. Not Predicated Off Threads Per Warp	29.7

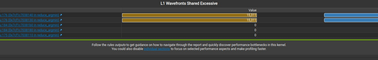
Warp State (All Cycles)

Instruction Statistics

Statistics of the executed low level assembly instructions (SASS). The instruction mix provides insight into the types and frequency of the executed instructions. A narrow mix of instruction types implies a dependency on few instruction pipelines, while others remain unused. Using multiple different types of instructions allows the hardware to execute more instructions concurrently, thus increasing the achieved instruction throughput.

Executed Instructions [inst]	2,352,128	Avg. Issued Instructions Per Scheduler	1.21
Issued Instructions [inst]	2,352,128	Avg. Issued Instructions Per Scheduler [inst]	13,798.31

Executed Instructions Mix



NVLink Topology

NVLink Topology diagram shows logical NVLink connections with (source) traffic throughput.

NVLink Tables

Detailed tables with properties for each NVLink.

Nvlink Admin

Non-uniform memory access (NUMA) affinities based on compute and memory distances for all GPUs.

Occupancy

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.

Block Size	128	Cluster Scheduling Policy	Spread
Registers Per Thread [register/thread]	16	Static Shared Memory Per Block [Kilobyte/block]	12.29
Grid Size	1,024	Dynamic Shared Memory Per Block [Kilobyte/block]	0
Threads	1,048,576	Driver Shared Memory Per Block [Kilobyte/block]	1.07
Waves Per SMs	20.48	Shared Memory Configuration Size [Kilobyte]	32.77
Hardware Limits	8	# Max Clusters	0

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 active in one cycle. 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 thread grids in combination with high kernel duration or high resource usage can benefit from higher occupancy.

Theoretical Occupancy [%]	100
Achieved Occupancy [%]	99.93
Achieved Active Warps Per SM [warp]	31.98

Achieved Occupancy The difference between calculated theoretical (100.0%) and measured achieved occupancy (99.9%) can be the result of warp scheduling overheads or workload imbalances during the kernel execution. Load imbalances can occur between warps within a block or well across blocks of the same kernel. See the Profiling Guide for more details on optimizing occupancy.

Impact of Varying Register Count Per Thread

Impact of Varying Block Size

Impact of Varying Shared Memory Usage Per Block

GPU and Memory Workload Distribution

Analysis of workload distribution in active cycles of SM, DRAM, GRAC+TLB, L1/TEX cache, and SYSMEM.

Average Active Cycles [cycle]	711,661.39	Average L1 Active Cycles [cycle]	711,661.39
Average L2 Active Cycles [cycle]	712,783.53	Average SMSP Active Cycles [cycle]	709,570.40
Total DRAM Elapsed Cycles [cycle]	2,146,304	Total L1/TEX Elapsed Cycles [cycle]	2,146,304
Total SM Elapsed Cycles [cycle]	26,836,480	Total L2 Elapsed Cycles [cycle]	24,208,608
Total SMSP Elapsed Cycles [cycle]	106,345,504	Total Global Elapsed Cycles [cycle]	49,546,624

Workload Distribution One or more SMSPs have a much lower number of active cycles than the average number of active cycles. Maximum instance value is 49.74% above the average, while the minimum instance value is 70.01% below the average.

Workload Distribution One or more L1/TEXs have a much higher number of active cycles than the average number of active cycles. Maximum instance value is 88.57% above the average, while the minimum instance value is 26.79% below the average.

Workload Distribution

	Average	Min	Max	Sum
SM Active Cycles	711,661.39	498,878	1,066,440	26,836,480
SMSP Active Cycles	709,570.40	498,878	1,066,440	106,345,504
L1 Active Cycles	711,661.39	498,878	1,066,440	26,836,480
L2 Active Cycles	712,783.53	75,196	1,055,904	3,005,070
DRAM Active Cycles	143,264	16,576	196,928	5,428,480

Source Counters

Source metrics, including branch efficiency and sampled warp stall reasons. Warp Stall Sampling metrics are periodically sampled over the kernel runtime. They indicate when warps were stalled and could not be scheduled. See the documentation for a description of all stall reasons. Only kernels compiled with -lineinfo will collect source level metrics.

Branch Instructions [inst]	2,352,128	Branch Efficiency [%]	99.94
Branch Instructions Ratio [%]	1.43	Avg. Divergent Branches	0

Uncoalesced Global Accesses This kernel has uncoalesced global accesses resulting in a total of 20023 excessive sectors (0% of the total 1,028,392 sectors). Check the L2 | Wavefronts Shared Excessive table for the primary source locations. The Memory Chart shows the sectors and utilization.

Shared Memory Bank Conflicts This kernel has uncoalesced shared accesses resulting in a total of 14219 bank conflicts (12.70% of the total 111950 wavefronts). Check the L1 | Wavefronts Shared Excessive table for the primary source locations. The Memory Chart shows the sectors and utilization. See an example on optimizing shared memory accesses.

L1 Wavefronts Shared Excessive

Location	Value	Value per
kernel_matrix_multiply_step_1(int, int, int,