	Result	Size	Time	Cycles	GPU	SM Frequency	Process	Attributes
Current	559 - brute_force	(4, 1, 1)x(1024, 1, 1)	11.94 ms	6,982,501	0 - Tesla T4	585.00 Mhz	[12774] exe	

Summary | Details | Source | Context | Comments | Raw | Session — Compare | Tools | View | Export

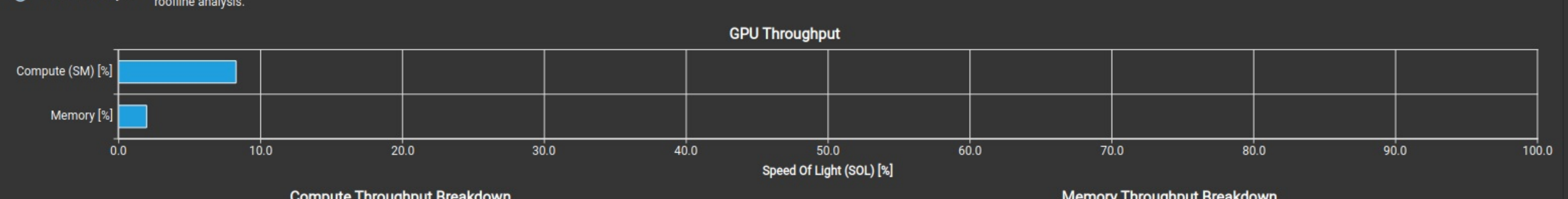
GPU Speed Of Light Throughput

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. High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.

Compute (SM) Throughput [%]	8.27	Duration [ms]	11.94
Memory Throughput [%]	1.97	Elapsed Cycles [cycle]	6,982,501
L1/TEX Cache Throughput [%]	20.40	SM Active Cycles [cycle]	674,879.12
L2 Cache Throughput [%]	0.00	SM Frequency [Mhz]	585.00
DRAM Throughput [%]	0.00	DRAM Frequency [Ghz]	5.00

Small Grid This kernel grid is too small to fill the available resources on this device, resulting in only 0.1 full waves across all SMs. Look at Launch Statistics for more details.

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

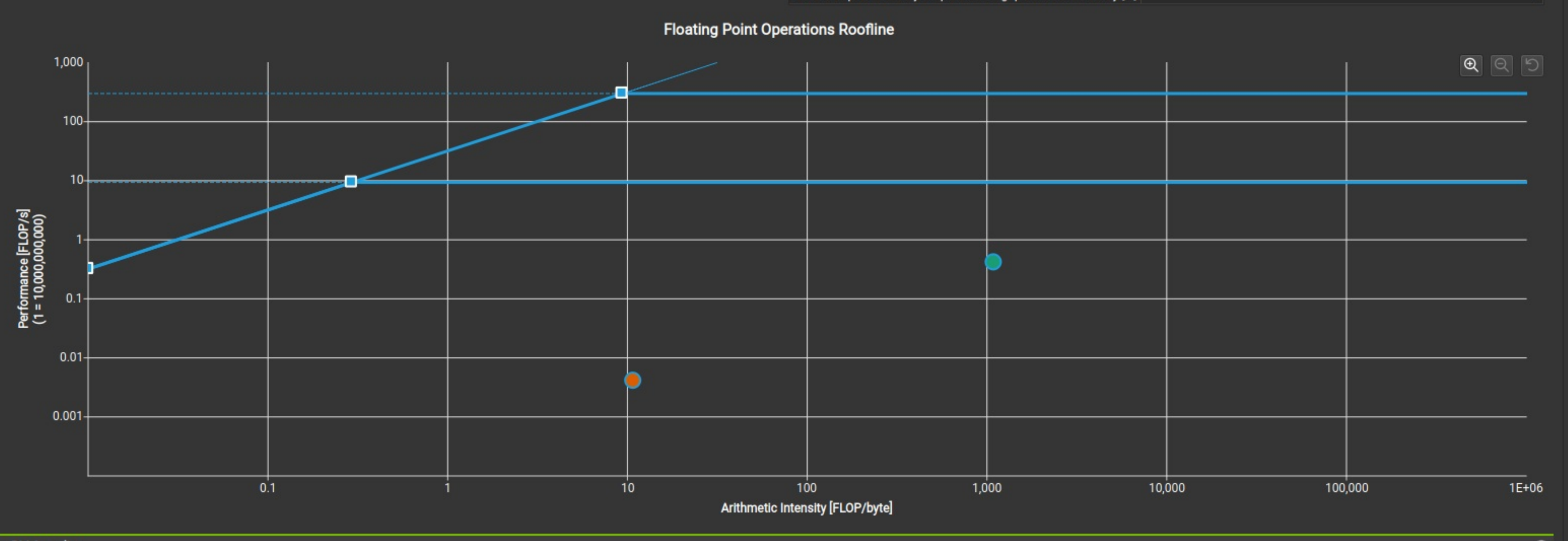


Compute Throughput Breakdown

SM: Pipe Alu Cycles Active [%]	8.27
SM: Issue Active [%]	7.16
SM: Inst Executed [%]	7.16
SM: Inst Executed Pipe Adu [%]	3.92
IDC: Request Cycles Active [%]	2.93
SM: Pipe Fma Cycles Active [%]	2.46
SM: Inst Executed Pipe Lsu [%]	1.97
SM: Mio Inst Issued [%]	1.96
SM: Mio Pq Write Cycles Active [%]	1.96
SM: Mio Pq Read Cycles Active [%]	1.94
SM: Inst Executed Pipe Cbu Pred On Any [%]	1.54
SM: Mio2rf Writeback Active [%]	0.97
SM: Pipe Fp64 Cycles Active [%]	0.06
SM: Inst Executed Pipe Uniform [%]	0.01
SM: Inst Executed Pipe Xu [%]	0.00
SM: Memory Throughput Internal Activity [%]	0
SM: Pipe Tensor Cycles Active [%]	0
SM: Pipe Shared Cycles Active [%]	0
SM: Instruction Throughput Internal Activity [%]	0
SM: Inst Executed Pipe Tex [%]	0
SM: Inst Executed Pipe Ipa [%]	0
SM: Inst Executed Pipe Fp16 [%]	0

Memory Throughput Breakdown

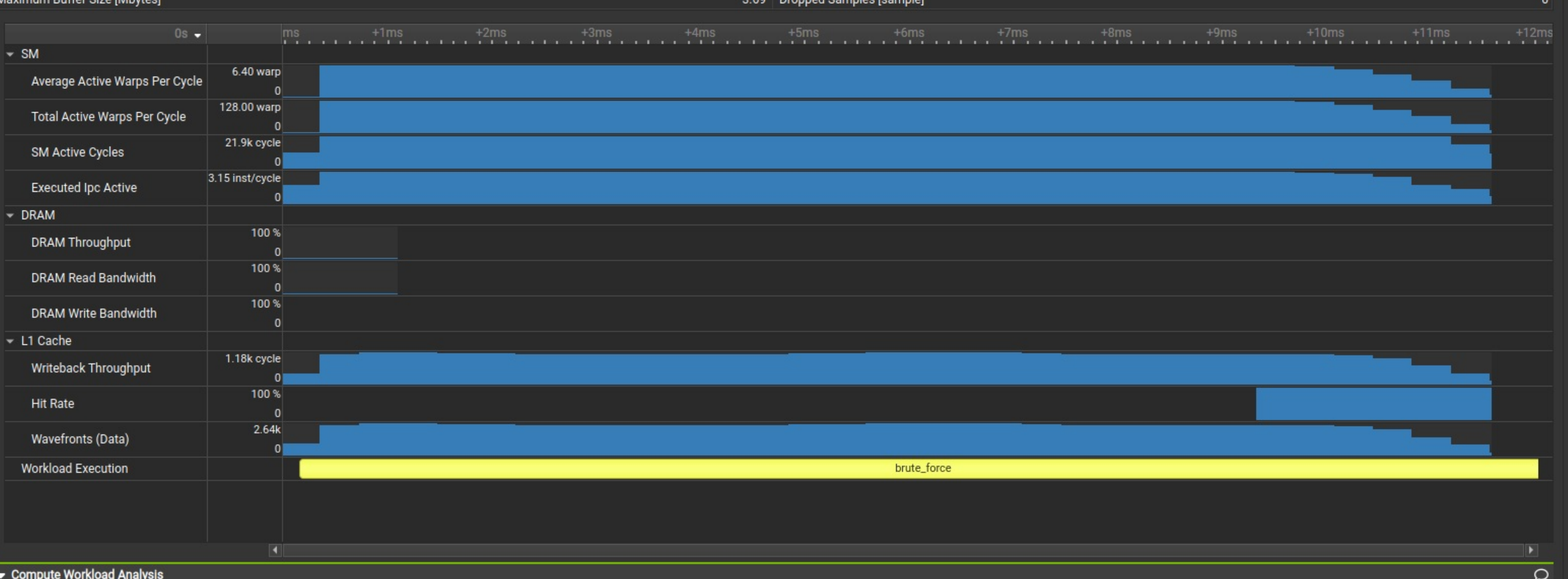
L1: Lsuin Requests [%]	1.97
L1: Data Pipe Lsu Wavefronts [%]	1.06
L1: Lsu Writeback Active [%]	0.47
L1: Data Bank Reads [%]	0.11
L1: Data Bank Writes [%]	0.06
L2: T Sectors [%]	0.00
DRAM: Cycles Active [%]	0.00
L2: Lts2xbar Cycles Active [%]	0.00
DRAM: Dram Sectors [%]	0.00
L2: Xbar2lts Cycles Active [%]	0.00
L1: M L1tex2xbar Req Cycles Active [%]	0.00
L2: D Sectors [%]	0.00
GPU: Compute Memory Access Throughput Internal Activity [%]	0.00
L2: T Tag Requests [%]	0.00
L2: D Sectors Fill Device [%]	0.00
L1: F Wavefronts [%]	0.00
L1: Texin Sm2tex Req Cycles Active [%]	0.00
L1: M Xbar2l1tex Read Sectors [%]	0
L1: Tex Writeback Active [%]	0
L2: D Atomic Input Cycles Active [%]	0
L2: D Sectors Fill Sysmem [%]	0
L1: Data Pipe Tex Wavefronts [%]	0
GPU: Compute Memory Request Throughput Internal Activity [%]	0



PM Sampling

Timeline view of PM metrics sampled 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]	320,000	# Pass Groups	1
Maximum Buffer Size [Mbytes]	3.69	Dropped Samples [sample]	0

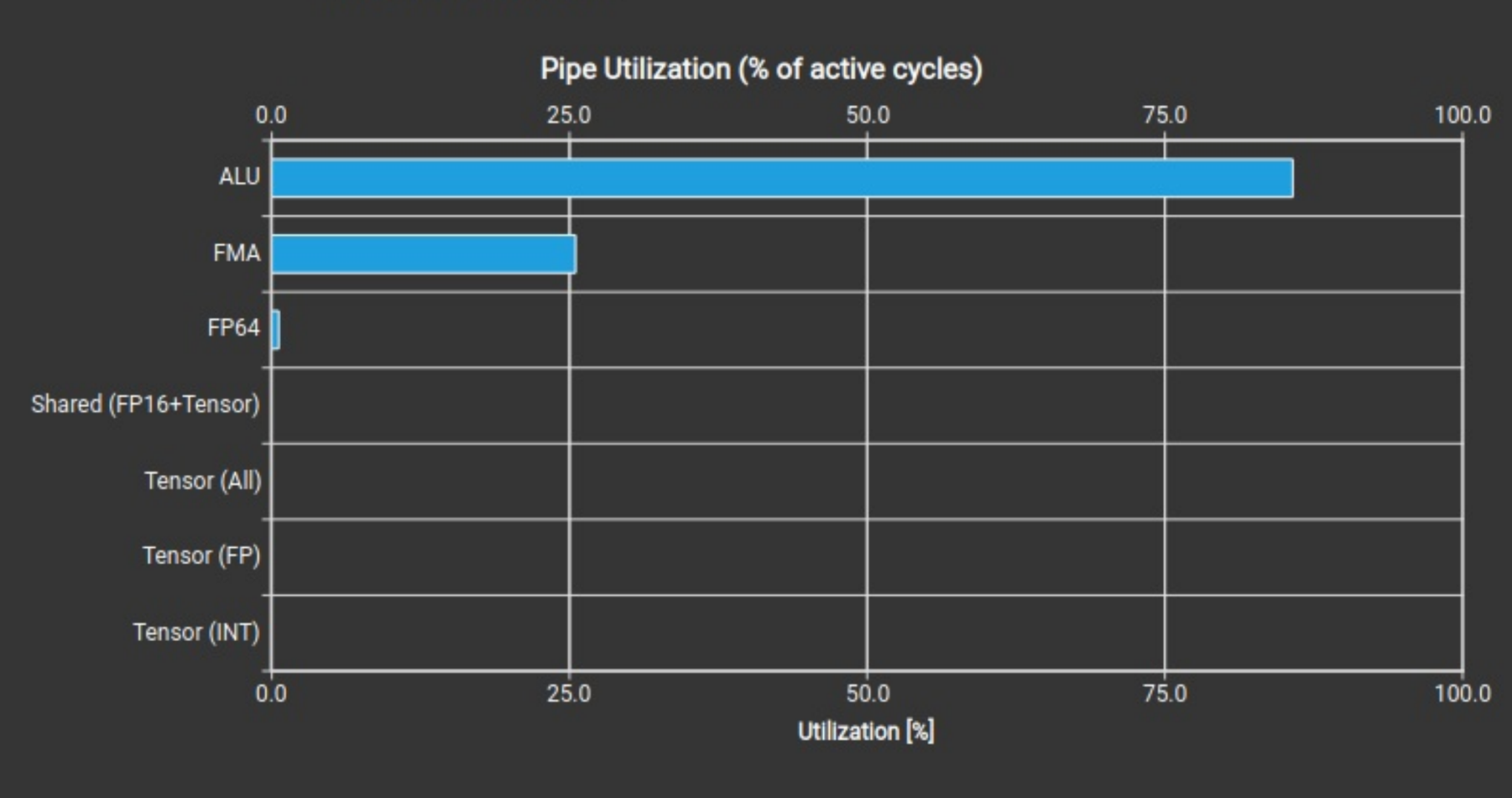


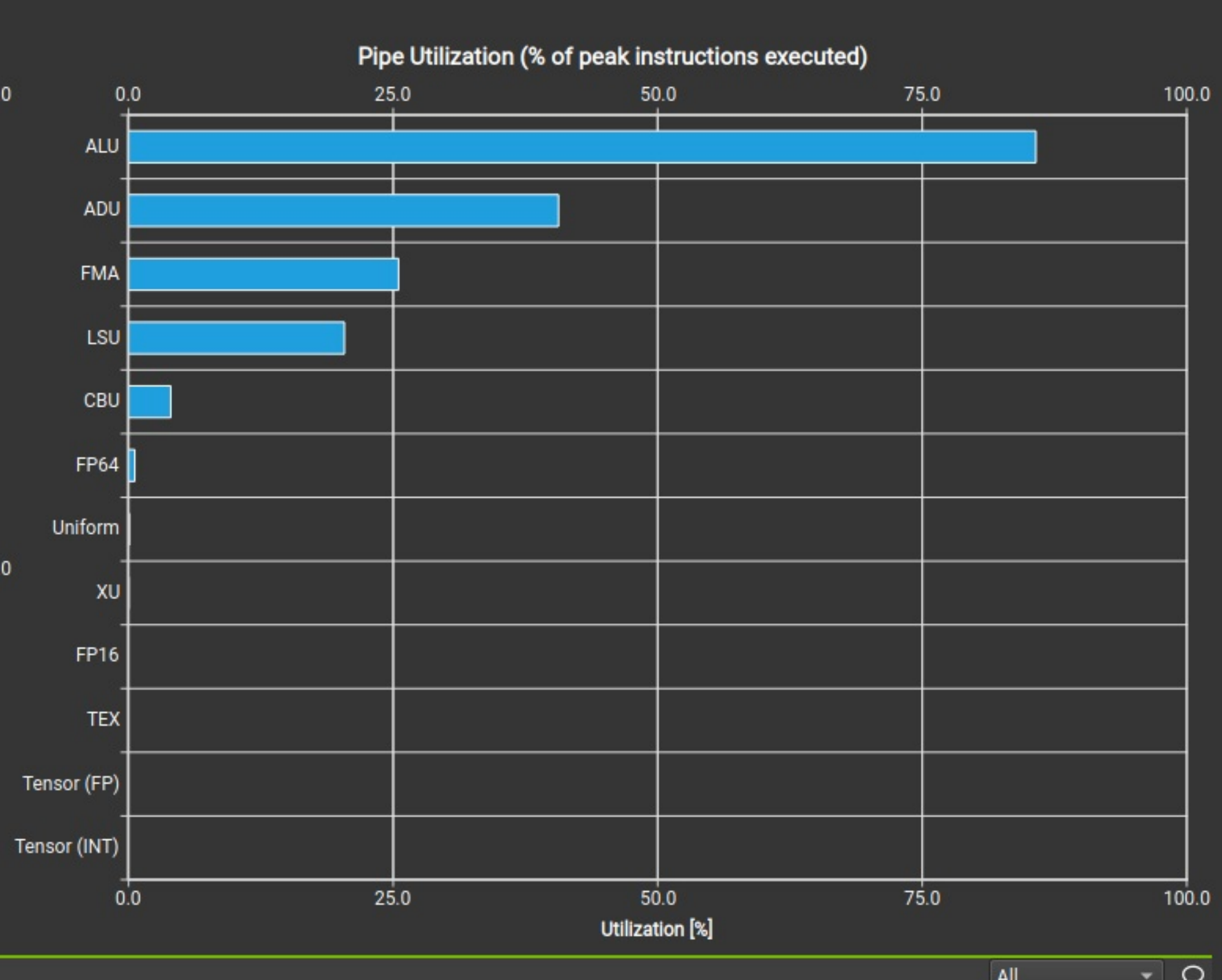
Compute Workload Analysis

Detailed analysis of the compute resources of the streaming multiprocessors (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]	0.29	SM Busy [%]	85.74
Executed Ipc Active [inst/cycle]	2.97	Issue Slots Busy [%]	74.26
Issued Ipc Active [inst/cycle]	2.97		

Very High Utilization ALU is the highest-utilized pipeline (85.7%) based on active cycles, taking into account the rates of its different instructions. It executes integer and logic operations. The pipeline is over-utilized and likely a performance bottleneck. Based on the number of executed instructions, the highest utilized pipeline (85.7%) is ALU. It executes integer and logic operations. Comparing the two, the overall pipeline utilization appears to be caused by frequent, low-latency instructions. See the Kernel Profiling Guide or hover over the pipeline name to understand the workloads handled by each pipeline. The Instruction Statistics section shows the mix of executed instructions in this kernel. Check the Warp State Statistics section for which reasons cause warps to stall.



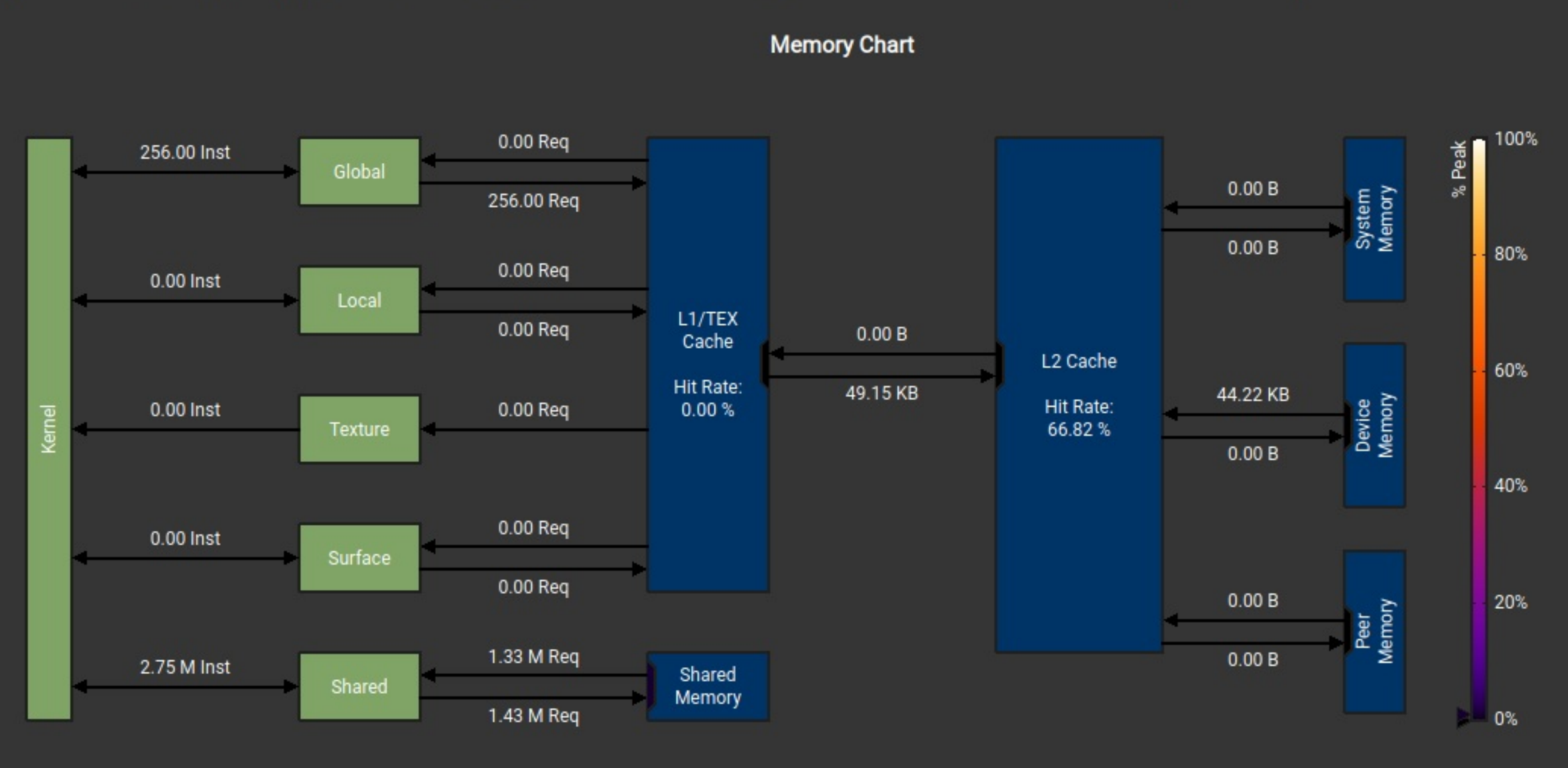


Memory Workload Analysis

Detailed analysis of the memory resources of the GPU. Memory can become a limiting factor for the overall kernel performance when fully utilizing the involved hardware units (Mem Busy), exhausting the available communication bandwidth between those units (Max Bandwidth), or by reaching the maximum throughput of issuing memory instructions (Mem Pipes Busy). Detailed chart of the memory units. Detailed tables with data for each memory unit.

Memory Throughput [Mbyte/s]	3.71	Mem Busy [%]	1.06
L1/TEX Hit Rate [%]	0	Max Bandwidth [%]	1.97
L2 Hit Rate [%]	66.82	Mem Pipes Busy [%]	3.92

Memory L2 Compression The optional metric lts__average_gcomp_input_sector_success_rate.pct could not be found. Collecting it as an additional metric could enable the rule to provide more guidance.



Shared Memory

	Instructions	Requests	Wavefronts	% Peak	Bank Conflicts
Shared Load	1,327,104	1,327,104	1,327,104	0.47	0
Shared Load Matrix	0	0			
Shared Store	1,425,408	1,425,408	1,639,989	0.59	0
Shared Atomic	0	0	0	0	0
Other	-	-	0	0	0
Total	2,752,512	2,752,512	2,967,093	1.06	0

L1/TEX Cache

	Instructions	Requests	Wavefronts	% Peak	Sectors	Sectors/Req	Hit Rate	Bytes	Sector Misses to L2	% Peak to L2	Returns to SM
Local Load	0	0	0	0	0	0	0	0	0	0	0
Global Load	0	0	0	0	0	0	0	0			
Surface Load	0	0	0	0	0	0	0	0	0	0	0
Texture Load	0	0	0	0	0	0	0	0	0	0	
Global Store	256	256	256	0.00	1,536	6	0	49,152	1,536	0.00	-
Local Store	0	0	0	0	0	0	0	0			-
Surface Store	0	0	0	0	0	0	0	0	0	0	-
Global Reduction	0	0	0	0	0	0	0	0	0	0	-
DSMEM Reduction						-	-		0	0	-
Surface Reduction	0	0	0	0	0	0	0	0	0	0	-
Global Atomic ALU	0	0	0	0	0	0	0	0	0	0	see above
Global Atomic CAS	0										
Surface Atomic ALU	0	0	0	0	0	0	0	0	0	0	see above
Surface Atomic CAS	0										
Loads	0	0	0	0	0	0	0	0	0	0	0
Stores	256	256	256	0.00	1,536	6	0	49,152	1,536	0.00	-
Atomics & Reductions	0	0	0	0	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 Misses to Peer
L1/TEX Load	0	0	0	0	0	0	0	0	0	0
L1/TEX Store	384	1,536	4	0.00	100	49,152	4,117,973.51	0	0	0
L1/TEX Atomic ALU	0	0	0	0	0	0	0	0	0	0
L1/TEX Atomic CAS	0	0	0	0	0	0	0	0	0	0
L1/TEX Reduction	0	0	0	0	0	0	0	0	0	0
L1/TEX Total	384	1,536	4	0.00	100	49,152	4,117,973.51	0	0	0
ECC Total	-	8	-	0.00	-	256	21,447.78	8	-	-
GPU Total	1,180	4,216	3.57	0.00	69.95	134,912	11,302,979.36	58	0	0

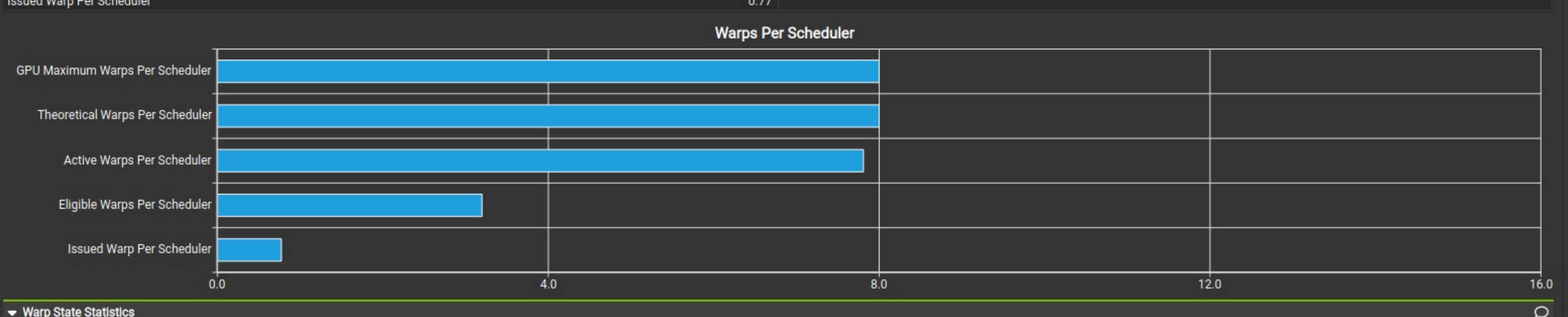
Device Memory

	Sectors	% Peak	Bytes	Throughput
Load	1,382	0.00	44,224	3,705,103.77
Store	0	0	0	0
Total	1,382	0.00	44,224	3,705,103.77

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 bound of warps in the pool (Theoretical Warps) is limited by the launch configuration. On every cycle each scheduler checks the state of the allocated warps in the pool (Active Warps). Active warps that are not stalled (Eligible Warps) are ready to issue their next instruction. From the set of eligible warps the scheduler selects a single warp from which to issue one or more instructions (Issued Warp). 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.

Active Warps Per Scheduler [warp]	7.81	No Eligible [%]	22.81
Eligible Warps Per Scheduler [warp]	3.20	One or More Eligible [%]	77.19
Issued Warp Per Scheduler	0.77		



Warp State Statistics

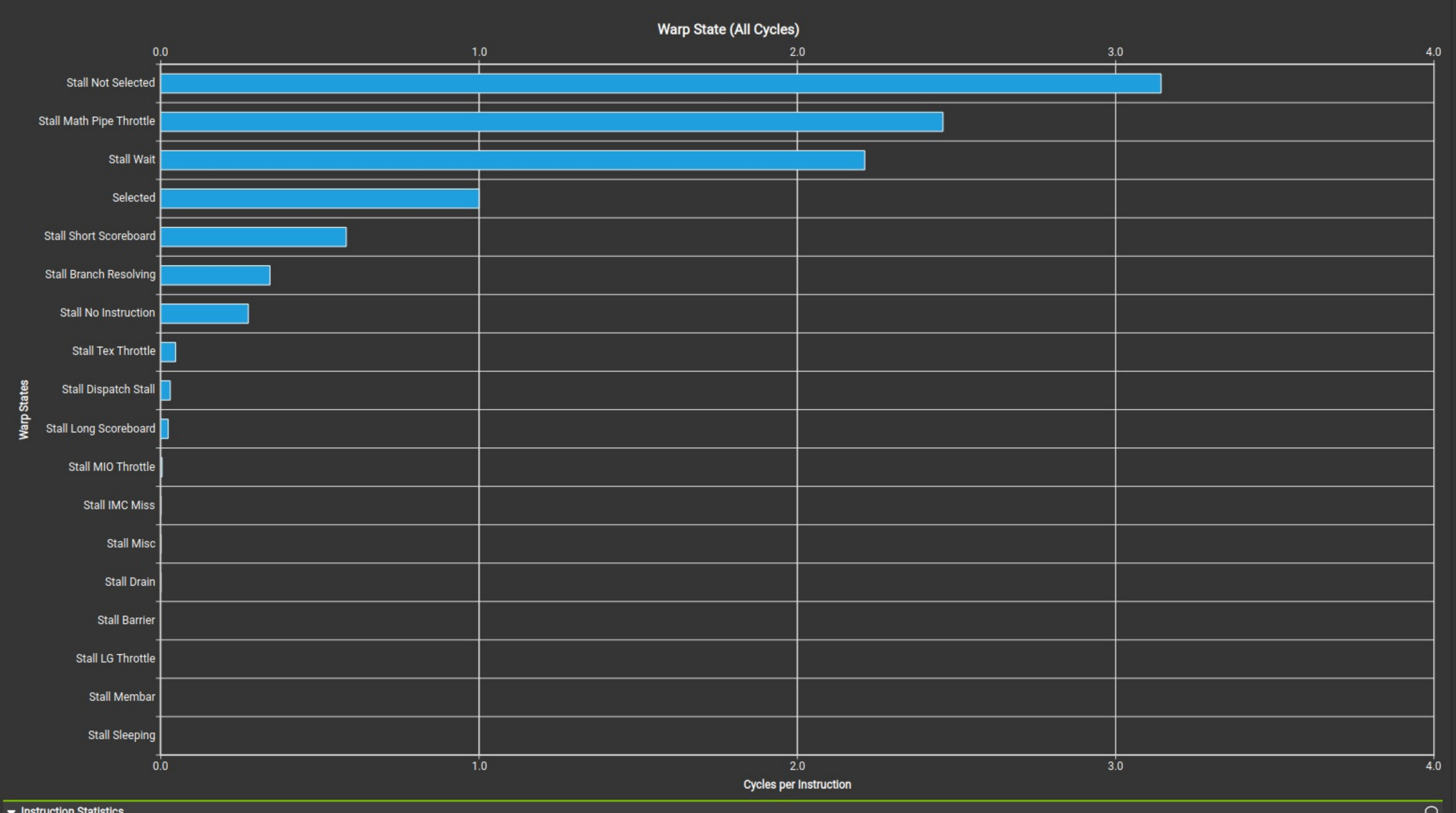
Analysis of the states in which all warps spent cycles during the kernel execution. The warp states describe a warp's readiness or inability to issue its next instruction. The warp cycles per instruction define the latency between two consecutive instructions. The higher the value, the more warp parallelism is required to hide this latency. For each warp state, the chart shows the average number of cycles spent in that state per issued instruction. Stalls are not always impacting the overall performance nor are they completely avoidable. Only focus on stall reasons if the schedulers fail to issue every cycle. When executing a kernel with mixed library and user code, these metrics show the combined values.

Warp Cycles Per Issued Instruction [cycle]	10.12	Avg. Active Threads Per Warp	23.25
Warp Cycles Per Executed Instruction [cycle]	10.12	Avg. Not Predicated Off Threads Per Warp	21.16

Not Selected Stalls Est. Local Speedup: 31.06% On average, each warp of this kernel spends 3.1 cycles being stalled waiting for the micro scheduler to select the warp to issue. Not selected warps are eligible warps that were not picked by the scheduler to issue that cycle as another warp was selected. A high number of not selected warps typically means you have sufficient warps to cover warp latencies and you may consider reducing the number of active warps to possibly increase cache coherence and data locality. This stall type represents about 31.1% of the total average of 10.1 cycles between issuing two instructions.

Warp Stall Check the Warp Stall Sampling (All Samples) table for the top stall locations in your source based on sampling data. The Kernel Profiling Guide provides more details on each stall reason.

Thread Divergence Est. Speedup: 2.80% 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 23.3 threads being active per cycle. This is further reduced to 21.2 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 when possible.



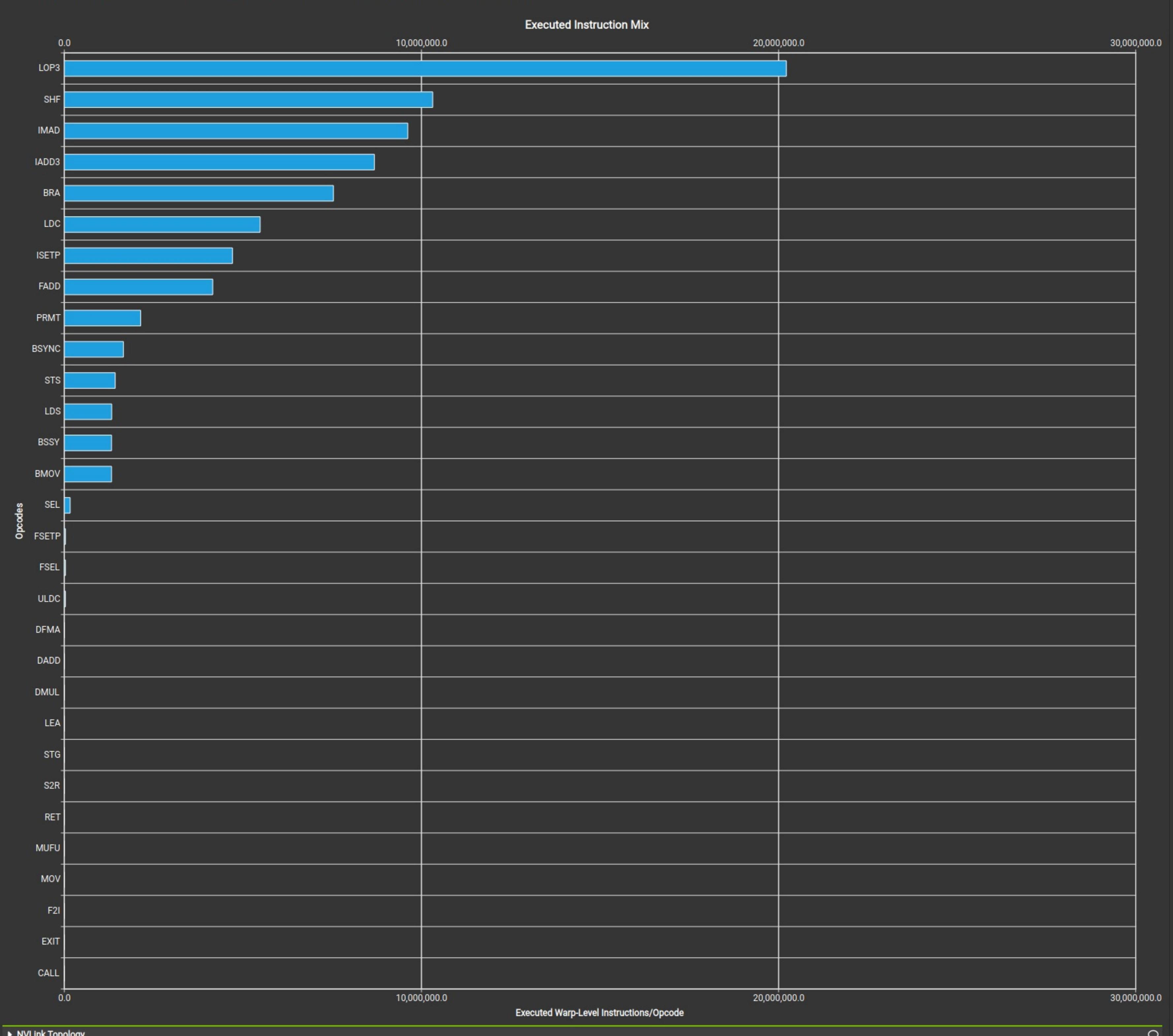
Instruction Statistics

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 pipelines allows hiding latencies and enables parallel execution. Note that 'Instructions/Opcode' and 'Executed Instructions' are measured differently and can diverge if cycles are spent in system calls.

Executed Instructions [inst]	80,182,400	Avg. Executed Instructions Per Scheduler [inst]	501,140
Issued Instructions [inst]	80,191,264	Avg. Issued Instructions Per Scheduler [inst]	501,195.40

FP32 Non-Fused Instructions Est. Speedup: 12.76% This kernel executes 0 fused and 4155904 non-fused FP32 instructions. By converting pairs of non-fused instructions to their fused, higher-throughput equivalent, the achieved FP32 performance could be increased by up to 50% (relative to its current performance). Check the Source page to identify where this kernel executes FP32 instructions.

FP64 Non-Fused Instructions Est. Speedup: 0.13% This kernel executes 5248 fused and 4480 non-fused FP64 instructions. By converting pairs of non-fused instructions to their fused, higher-throughput equivalent, the achieved FP64 performance could be increased by up to 23% (relative to its current performance). Check the Source page to identify where this kernel executes FP64 instructions.



NVLink Topology

NVLink Topology diagram shows logical NVLink connections with transmit/receive throughput.

NVLink Tables

Detailed tables with properties for each NVLink.

NUMA Affinity

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

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.

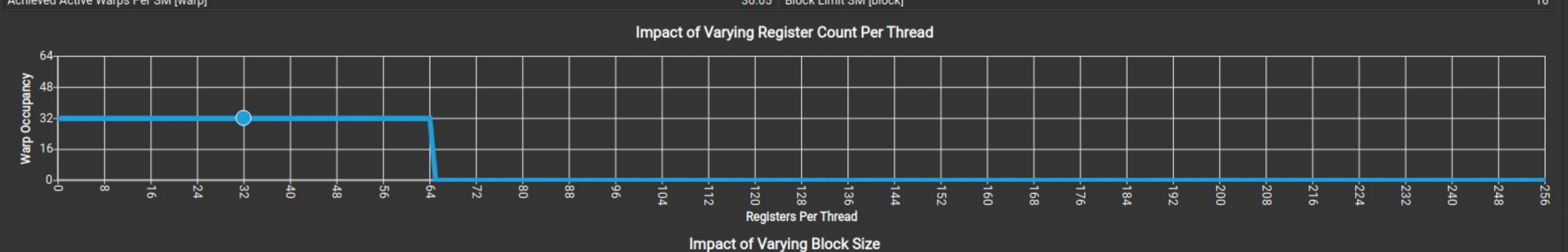
Grid Size	4	Function Cache Configuration	CachePreferNone
Registers Per Thread [register/thread]	32	Static Shared Memory Per Block [byte/block]	0
Block Size	1,024	Dynamic Shared Memory Per Block [Kbyte/block]	3.07
Threads [thread]	4,096	Driver Shared Memory Per Block [byte/block]	0
Waves Per SM	0.10	Shared Memory Configuration Size [Kbyte]	32.77
Uses Green Context	0	# SMs [SM]	40

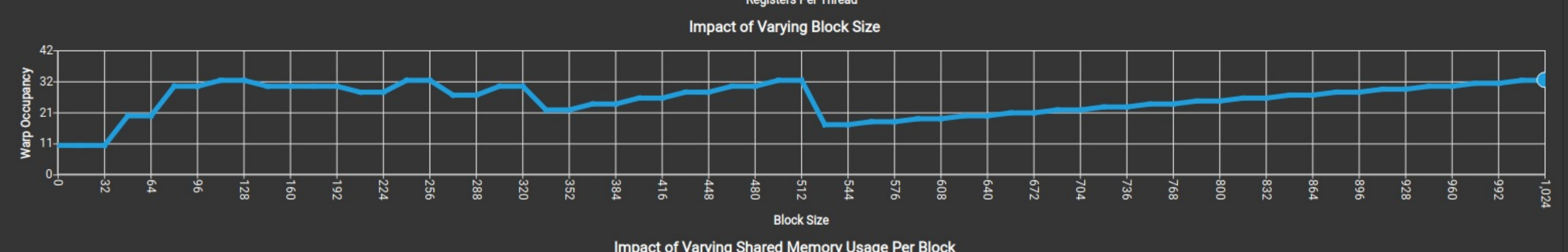
Small Grid Est. Speedup: 90.00% The grid for this launch is configured to execute only 4 blocks, which is less than the GPU's 40 multiprocessors. This can underutilize some multiprocessors. If you do not intend to execute this kernel concurrently with other workloads, consider reducing the block size to have at least one block per multiprocessor or increase the size of the grid to fully utilize the available hardware resources. See the Hardware Model description for more details on launch configurations.

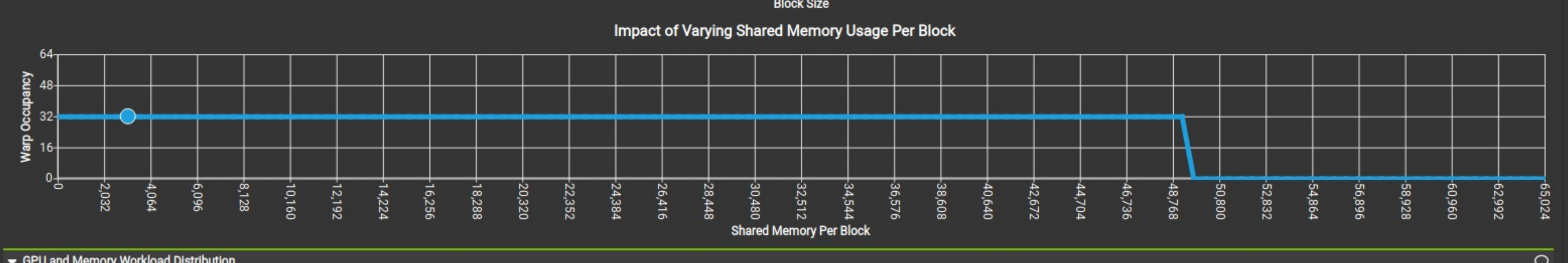
Occupancy

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 [%]	100	Block Limit Registers [block]	2
Theoretical Active Warps per SM [warp]	32	Block Limit Shared Mem [block]	10
Achieved Occupancy [%]	93.91	Block Limit Warps [block]	1
Achieved Active Warps Per SM [warp]	30.05	Block Limit SM [block]	16







GPU and Memory Workload Distribution

Analysis of workload distribution in active cycles of SM, SMP, SMSP, L1 & L2 caches, and DRAM

Average SM Active Cycles [cycle]	674,879.12	Average L1 Active Cycles [cycle]	674,879.12
Average L2 Active Cycles [cycle]	495.78	Average SMSP Active Cycles [cycle]	649,278.53
Average DRAM Active Cycles [cycle]	691	Total SM Elapsed Cycles [cycle]	279,873,744
Total L1 Elapsed Cycles [cycle]	279,873,744	Total L2 Elapsed Cycles [cycle]	326,566,240
Total SMSP Elapsed Cycles [cycle]	1,119,494,976	Total DRAM Elapsed Cycles [cycle]	477,454,336

SMs Workload Imbalance Est. Speedup: 8.71% One or more SMs have a much lower number of active cycles than the average number of active cycles. Maximum instance value is 90.33% above the average, while the minimum instance value is 100.00% below the average.

SMSPs Workload Imbalance Est. Speedup: 8.42% One or more SMSPs have a much lower number of active cycles than the average number of active cycles. Maximum instance value is 90.71% above the average, while the minimum instance value is 100.00% below the average.

L1 Slices Workload Imbalance Est. Speedup: 8.71% One or more L1 Slices have a much lower number of active cycles than the average number of active cycles. Maximum instance value is 90.33% above the average, while the minimum instance value is 100.00% below the average.

Workload Distribution

	Average	Min	Max	Sum
SM Active Cycles	674,879.12	0	6,980,318	26,995,165
SMSP Active Cycles	649,278.53	0	6,987,722	103,884,564
L1 Active Cycles	674,879.12	0	6,980,318	26,995,165
L2 Active Cycles	495.78	388	1,231	15,865
DRAM Active Cycles	691	528	936	5,528

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 couldn't be scheduled. See the documentation for a description of all stall reasons. Only focus on stalls if the schedulers fail to issue every cycle.

Branch Instructions [inst]	9,191,424	Branch Efficiency [%]	95.58
Branch Instructions Ratio [%]	0.11	Avg. Divergent Branches	2,080

Follow the *rules outputs* to get guidance on how to navigate through the report and quickly discover performance bottlenecks in this kernel.
You could also disable individual sections to focus on selected performance aspects and make profiling faster.