Result	Size	Time	Cycles	GPU	SM Frequency	Process	Attributes
Current: 680 - brute_force_AL	(1024, 1, 1)x(1024, 1, 1)	1.10 ms	642,480	0 - Tesla T4	584.97 Mhz	[14702] exe	

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

GPU Speed Of Light Throughput

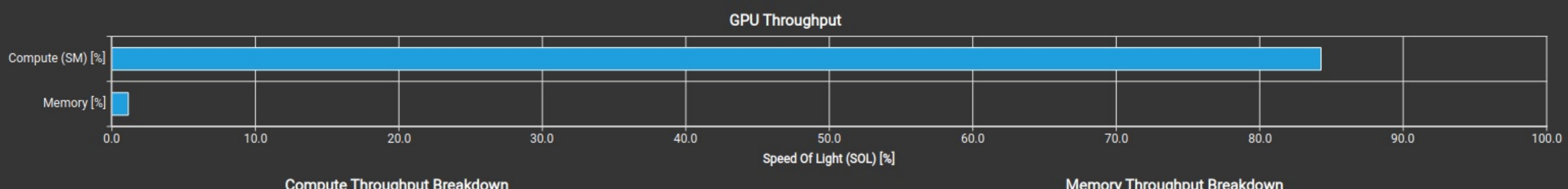
All

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 [%]	84.28	Duration [ms]	1.10
Memory Throughput [%]	1.14	Elapsed Cycles [cycle]	642,480
L1/TEX Cache Throughput [%]	1.02	SM Active Cycles [cycle]	623,261.55
L2 Cache Throughput [%]	0.45	SM Frequency [Mhz]	584.97
DRAM Throughput [%]	1.14	DRAM Frequency [Ghz]	4.99

High Throughput The kernel is utilizing greater than 80.0% of the available compute or memory performance of the device. To further improve performance, work will likely need to be shifted from the most utilized to another unit. Start by analyzing workloads in the Compute Workload Analysis section.

Roofline Analysis The ratio of peak float (fp32) to double (fp64) performance on this device is 32:1. The kernel achieved 2% of this device's fp32 peak performance and 0% of its fp64 peak performance. See the Kernel Profiling Guide for more details on roofline analysis.

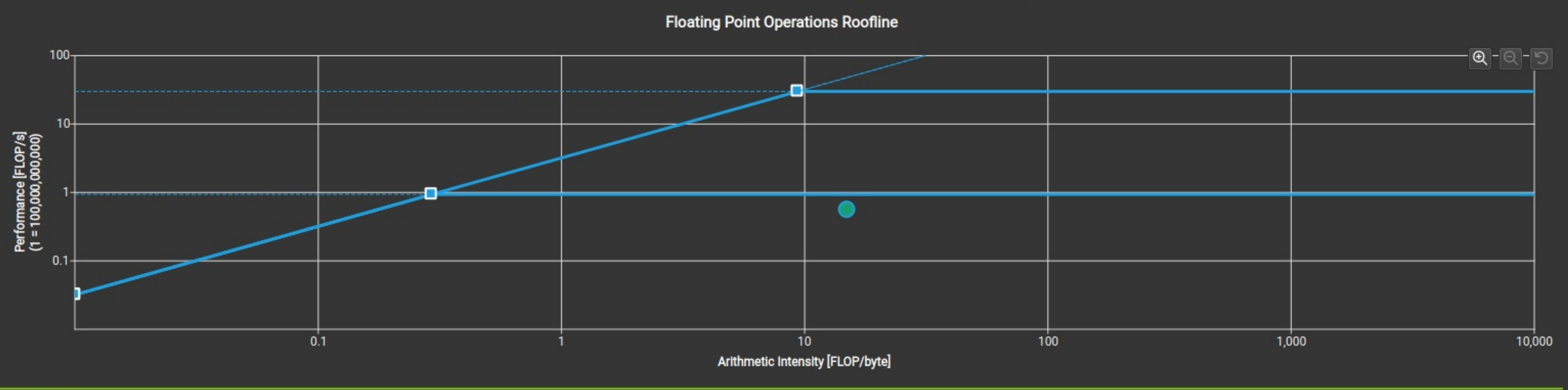


Compute Throughput Breakdown

SM: Pipe Alu Cycles Active [%]	84.28
SM: Issue Active [%]	62.25
SM: Inst Executed [%]	62.24
SM: Inst Executed Pipe Adu [%]	38.53
IDC: Request Cycles Active [%]	22.02
SM: Pipe Fma Cycles Active [%]	17.41
SM: Mio Inst Issued [%]	13.01
SM: Mio Pq Read Cycles Active [%]	11.14
SM: Mio Pq Write Cycles Active [%]	11.14
SM: Inst Executed Pipe Cbu Pred On Any [%]	8.42
SM: Mio2rf Writeback Active [%]	5.63
SM: Inst Executed Pipe Lsu [%]	0.51
SM: Inst Executed Pipe Uniform [%]	0.06
SM: Memory Throughput Internal Activity [%]	0
SM: Pipe Tensor Cycles Active [%]	0
SM: Pipe Shared Cycles Active [%]	0
SM: Pipe Fp64 Cycles Active [%]	0
SM: Instruction Throughput Internal Activity [%]	0
SM: Inst Executed Pipe Xu [%]	0
SM: Inst Executed Pipe Tex [%]	0
SM: Inst Executed Pipe Ipa [%]	0
SM: Inst Executed Pipe Fp16 [%]	0

Memory Throughput Breakdown

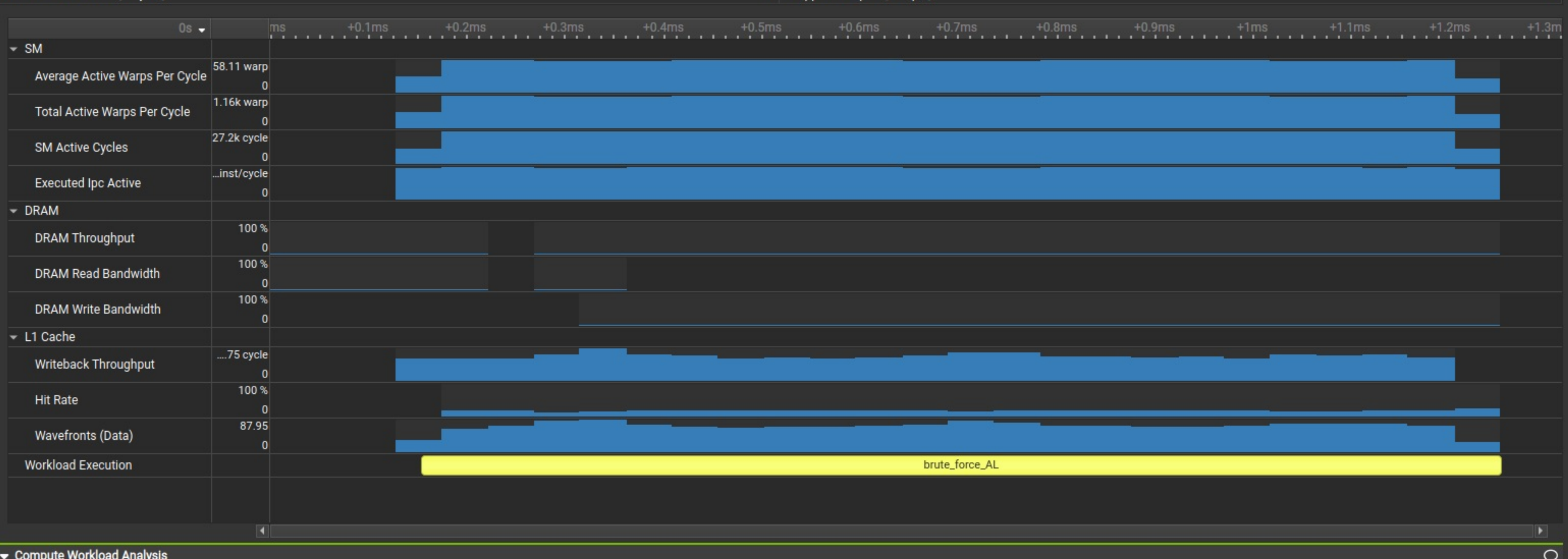
DRAM: Cycles Active [%]	1.14
DRAM: Dram Sectors [%]	0.84
L1: M L1tex2xbar Req Cycles Active [%]	0.51
L1: Lsuin Requests [%]	0.51
L2: T Sectors [%]	0.45
L2: Xbar2lts Cycles Active [%]	0.44
L1: Data Pipe Lsu Wavefronts [%]	0.26
L2: D Sectors [%]	0.19
L1: Lsu Writeback Active [%]	0.13
L2: T Tag Requests [%]	0.11
GPU: Compute Memory Access Throughput Internal Activity [%]	0.11
L1: Data Bank Reads [%]	0.06
L1: Data Bank Writes [%]	0.06
L2: Lts2xbar Cycles Active [%]	0.01
L1: Texin Sm2tex Req Cycles Active [%]	0.00
L1: F Wavefronts [%]	0.00
L2: D Sectors Fill Device [%]	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]	40,000	# Pass Groups	1
Maximum Buffer Size [Mbytes]	3	Dropped Samples [sample]	0

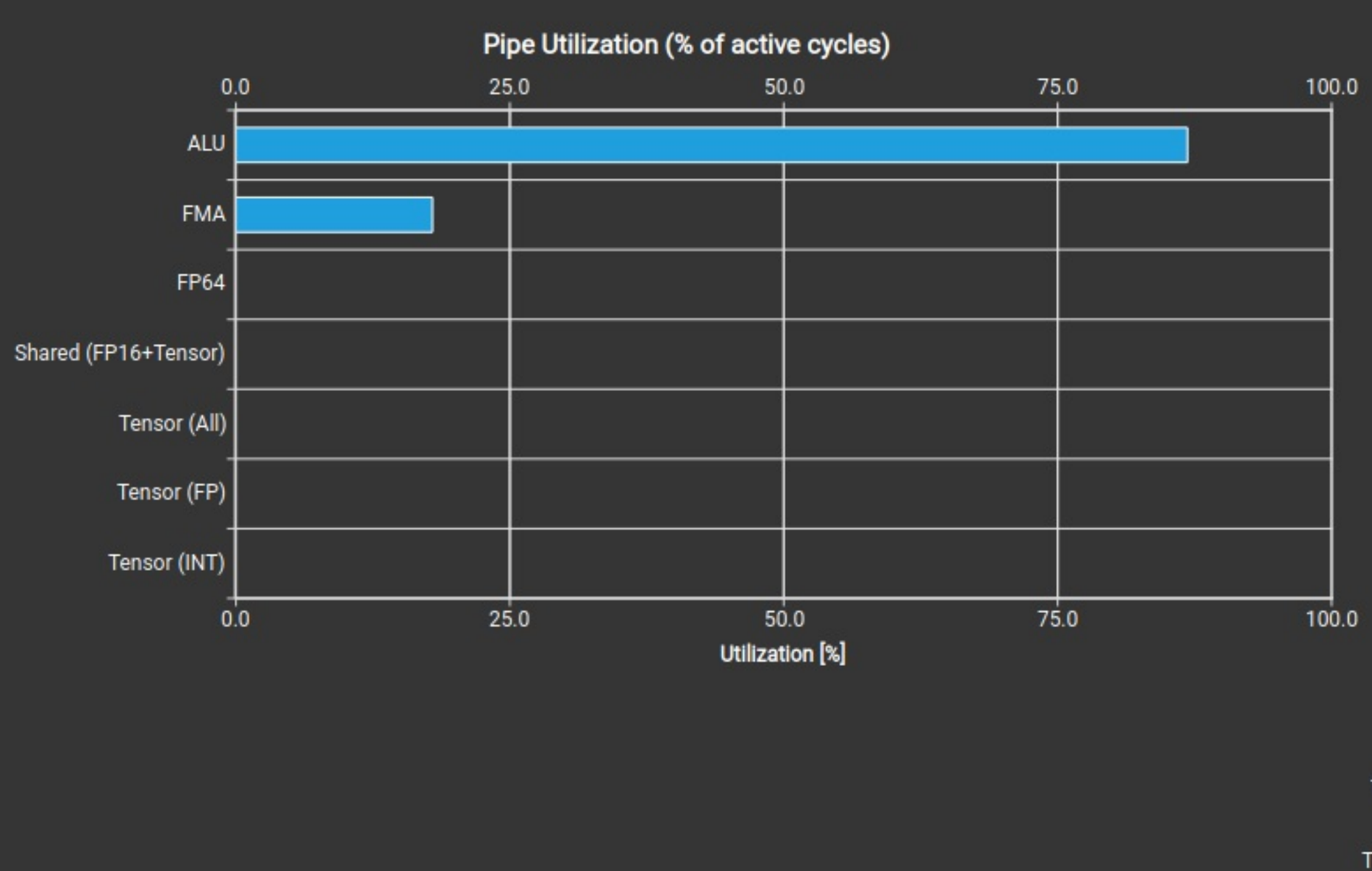


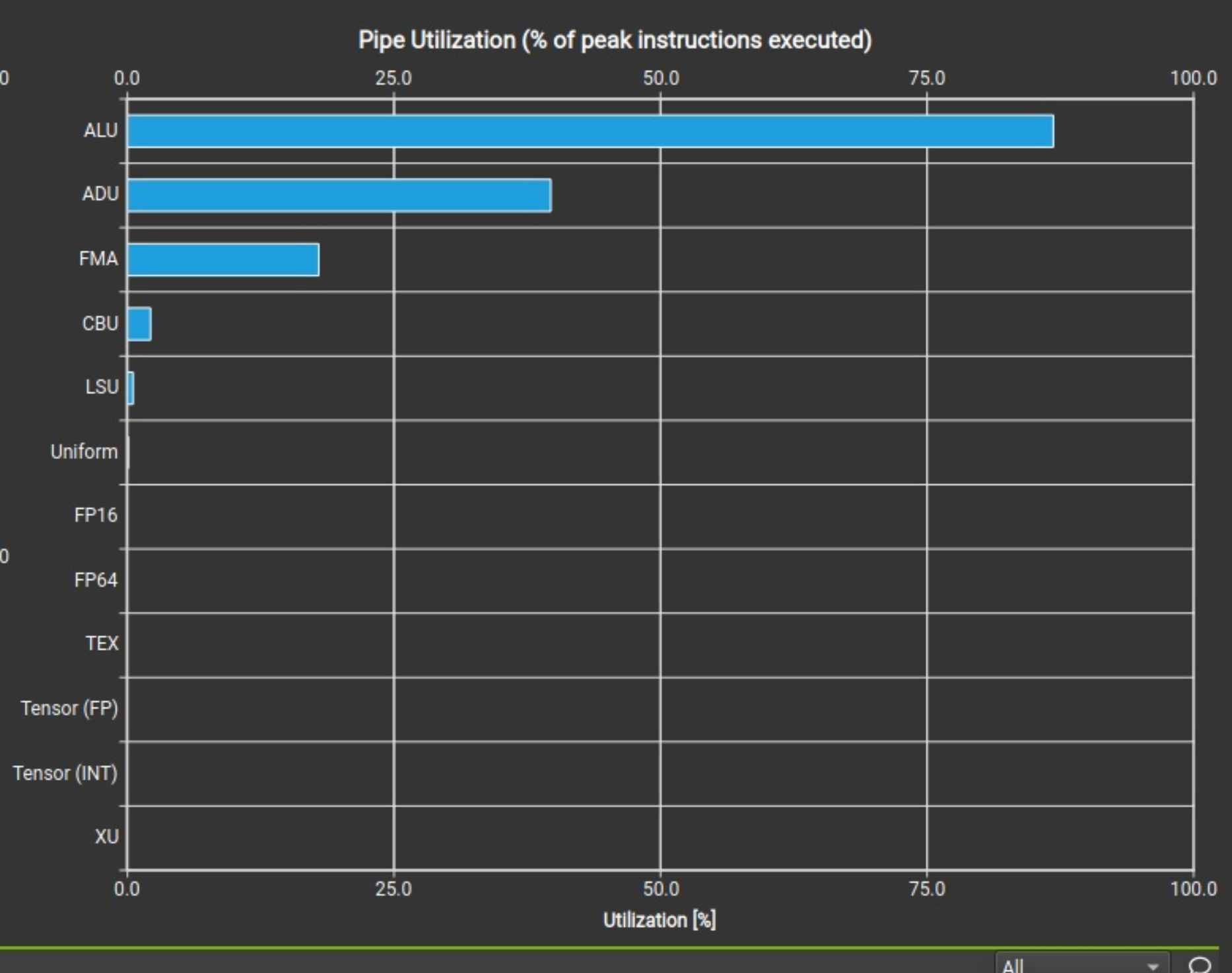
Compute Workload Analysis

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]	2.49	SM Busy [%]	86.85
Executed Ipc Active [inst/cycle]	2.57	Issue Slots Busy [%]	64.15
Issued Ipc Active [inst/cycle]	2.57		

Very High Utilization ALU is the highest-utilized pipeline (86.8%) 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 (86.8%) 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

All

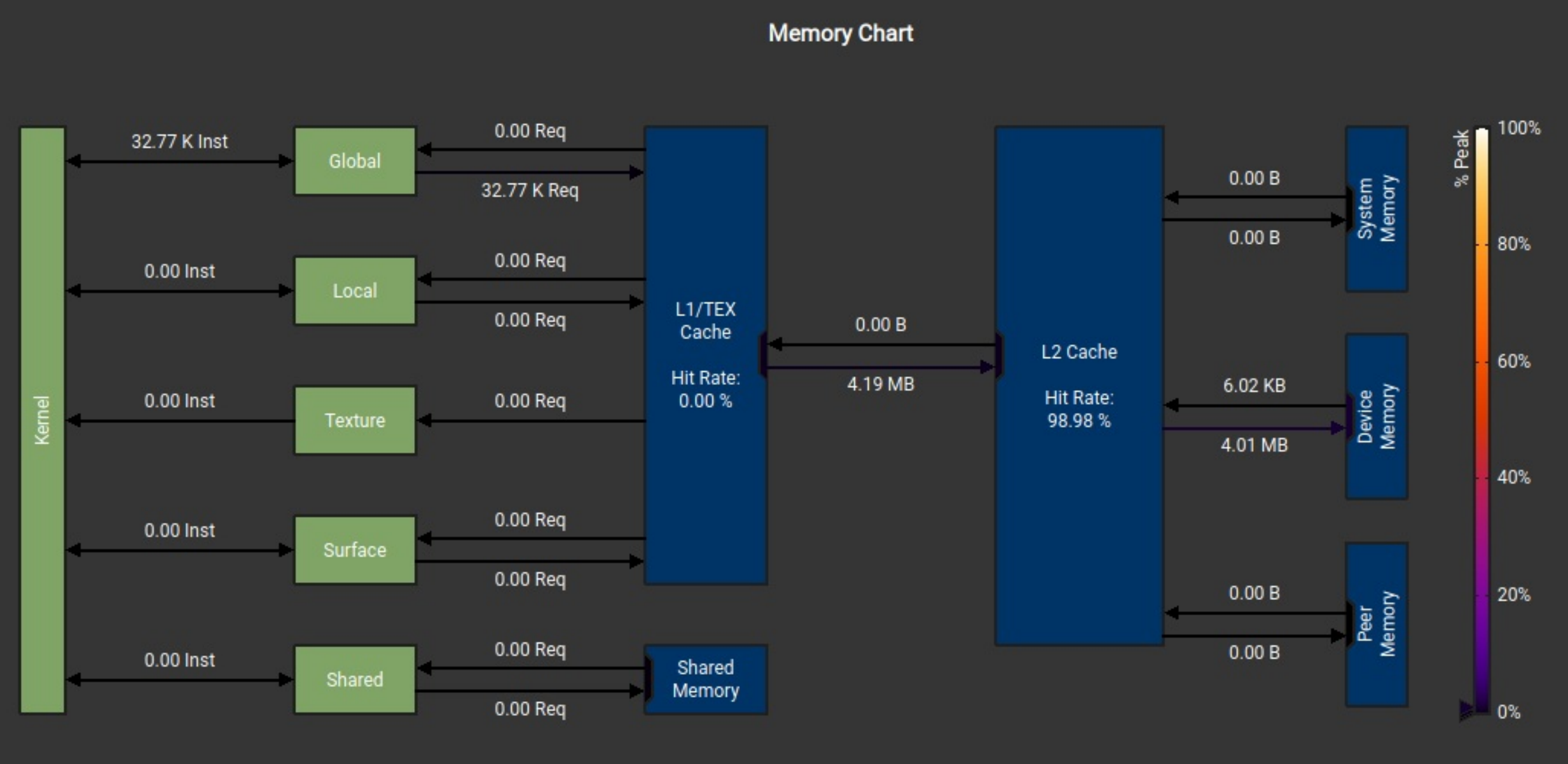
Detailed analysis of the memory resources of the GPU. Memory can become a limiting factor for the overall kernel performance when fully utilizing the involved hardware units (Mem Busy), exhausting the available communication bandwidth between those units (Max Bandwidth), or by reaching the maximum throughput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

Memory Throughput [Gbyte/s]	3.66	Mem Busy [%]	0.45
L1/TEX Hit Rate [%]	0	Max Bandwidth [%]	1.14
L2 Hit Rate [%]	98.98	Mem Pipes Busy [%]	38.53

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.

Memory Chart

Show As: Transfer Size



Shared Memory

	Instructions	Requests	Wavefronts	% Peak	Bank Conflicts
Shared Load	0	0	0	0	0
Shared Load Matrix	0	0			
Shared Store	0	0	0	0	0
Shared Atomic	0	0	0	0	0
Other	-	-	32,768	0.13	0
Total	0	0	32,768	0.13	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	32,768	32,768	32,768	0.13	131,072	4	0	4,194,304	131,072	0.51	-
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	32,768	32,768	32,768	0.13	131,072	4	0	4,194,304	131,072	0.51	-
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	32,768	131,072	4	0.44	100	4,194,304	3,818,891,672.98	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	32,768	131,072	4	0.44	100	4,194,304	3,818,891,672.98	0	0	0
ECC Total	-	8	-	0.00	-	256	233,086.65	8	-	-
GPU Total	34,288	134,988	3.94	0.45	99.97	4,319,616	3,932,987,588.14	404	0	0

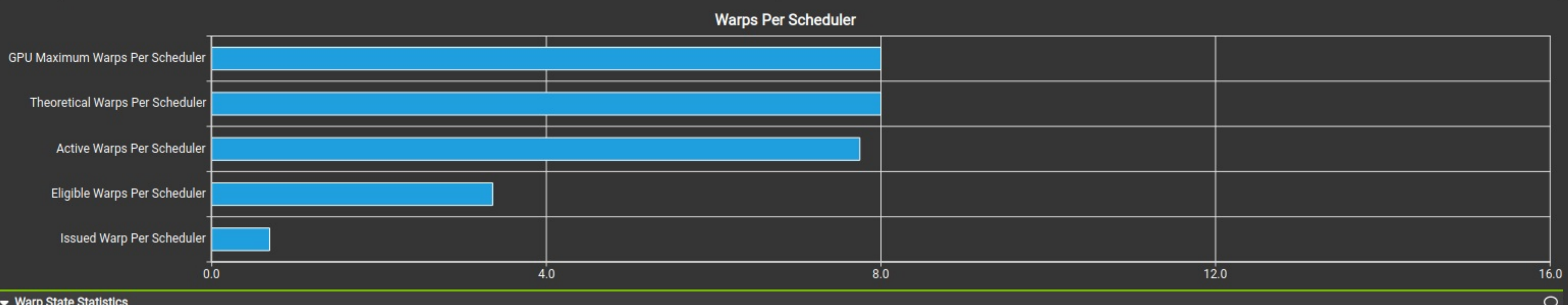
Device Memory

	Sectors	% Peak	Bytes	Throughput
Load	188	0.00	6,016	5,477,536.27
Store	125,282	1.14	4,009,024	3,650,195,210.07
Total	125,470	1.14	4,015,040	3,655,672,746.34

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]	7.75	No Eligible [%]	30.79
Eligible Warps Per Scheduler [warp]	3.36	One or More Eligible [%]	69.21
Issued Warp Per Scheduler	0.69		



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.

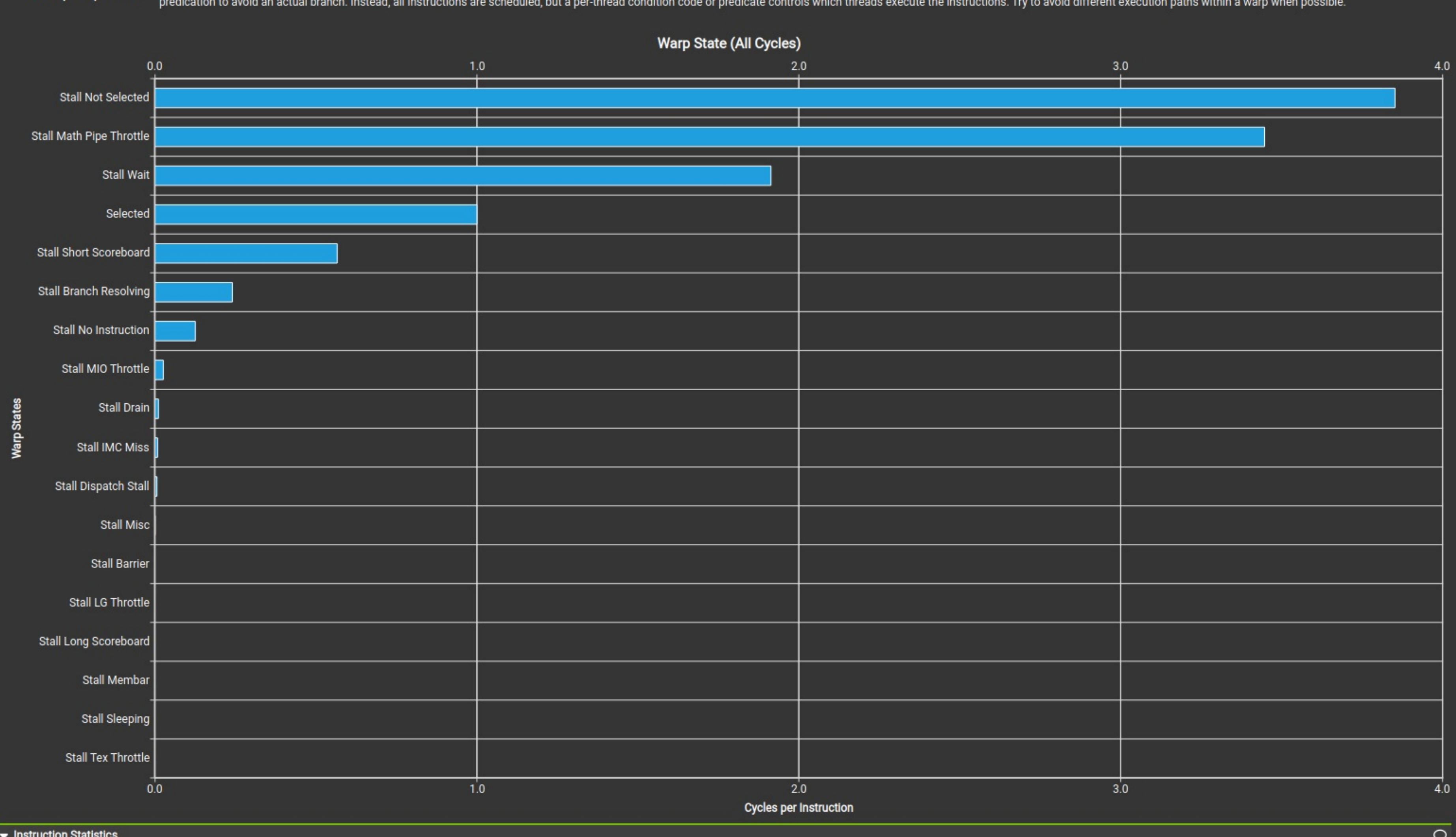
Warp Cycles Per Issued Instruction [cycle]	11.19	Avg. Active Threads Per Warp	24.15
Warp Cycles Per Executed Instruction [cycle]	11.20	Avg. Not Predicated Off Threads Per Warp	21.70

Not Selected Stalls Est. Local Speedup: 34.41% On average, each warp of this kernel spends 3.9 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 34.4% of the total average of 11.2 cycles between issuing two instructions.

Math Pipe Throttle Stalls Est. Local Speedup: 30.79% On average, each warp of this kernel spends 3.4 cycles being stalled waiting for the execution pipe to be available. This stall occurs when all active warps execute their next instruction on a specific, oversubscribed math pipeline. Try to increase the number of active warps to hide the existent latency or try changing the instruction mix to utilize all available pipelines in a more balanced way. This stall type represents about 30.8% of the total average of 11.2 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: 27.11% 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 24.1 threads being active per cycle. This is further reduced to 21.7 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 when possible.

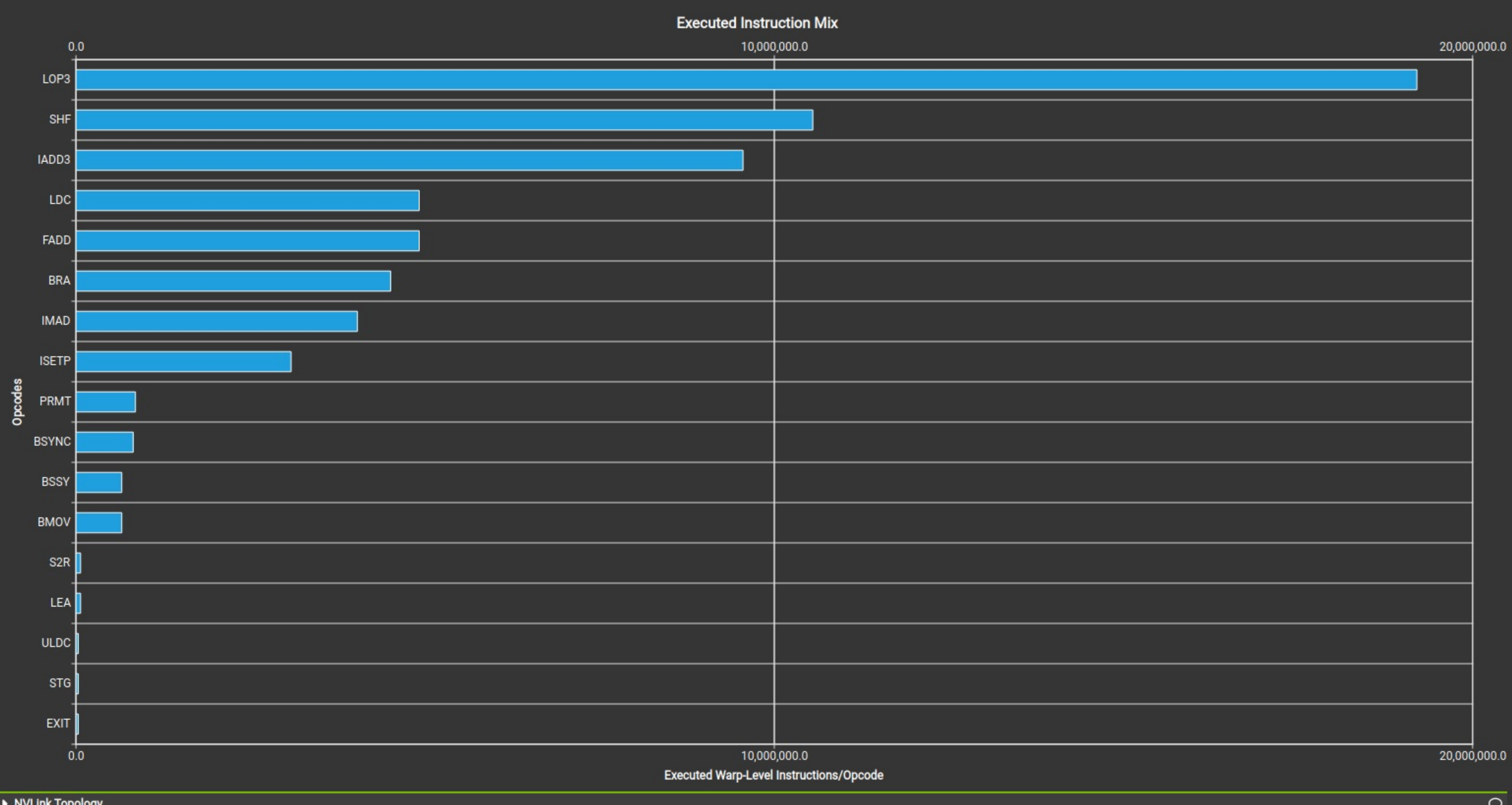


Instruction Statistics

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]	63,963,136	Avg. Executed Instructions Per Scheduler [inst]	399,769.60
Issued Instructions [inst]	63,973,086	Avg. Issued Instructions Per Scheduler [inst]	399,831.79

FP32 Non-Fused Instructions Est. Speedup: 8.97% This kernel executes 0 fused and 4915200 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.



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.

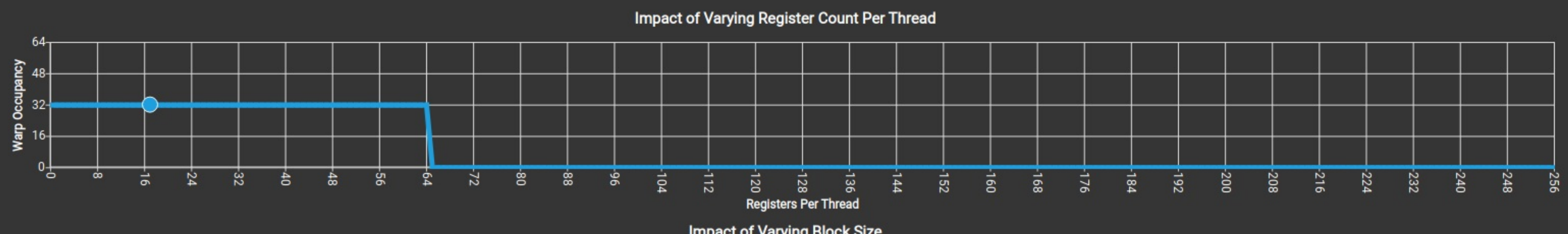
Grid Size	1,024	Function Cache Configuration	CachePreferNone
Registers Per Thread [register/thread]	17	Static Shared Memory Per Block [byte/block]	0
Block Size	1,024	Dynamic Shared Memory Per Block [Kbyte/block]	3.07
Threads [thread]	1,048,576	Driver Shared Memory Per Block [byte/block]	0
Waves Per SM	25.60	Shared Memory Configuration Size [Kbyte]	32.77
Uses Green Context	0	# SMs [SM]	40

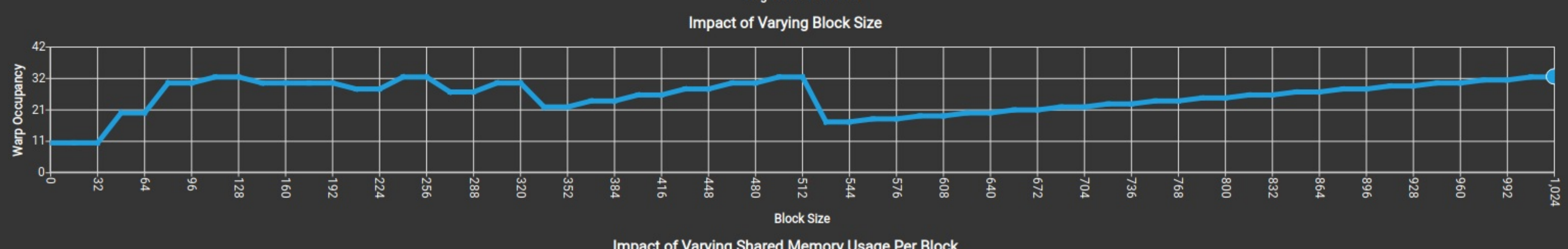
Occupancy

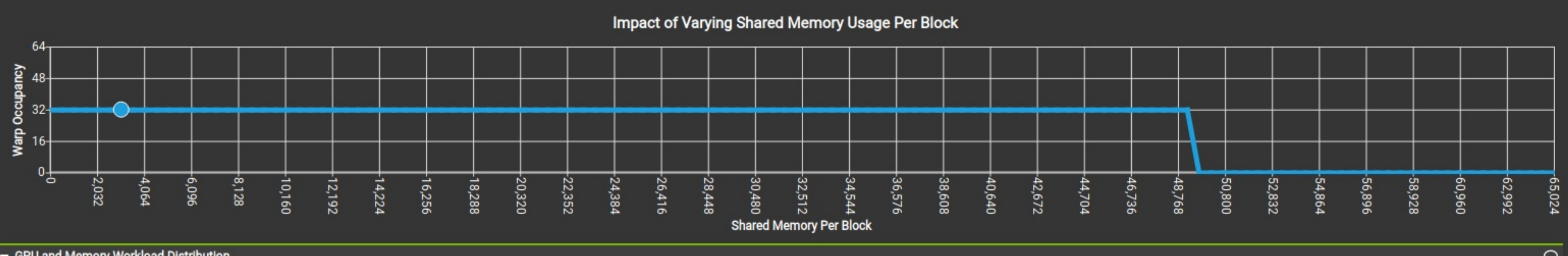
Occupancy is the ratio of the number of active warps per multiprocessor to the maximum number of possible active warps. Another way to view occupancy is the percentage of the hardware's ability to process warps that is actively in use. Higher occupancy does not always result in higher performance, however, 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 [%]	89.84	Block Limit Warps [block]	1
Achieved Active Warps Per SM [warp]	28.75	Block Limit SM [block]	16

Achieved Occupancy Est. Local Speedup: 10.16% The difference between calculated theoretical (100.0%) and measured achieved occupancy (89.8%) 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.







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]	623,261.55	Average L1 Active Cycles [cycle]	623,261.55
Average L2 Active Cycles [cycle]	44,179.50	Average SMSP Active Cycles [cycle]	577,681.41
Average DRAM Active Cycles [cycle]	62,735	Total SM Elapsed Cycles [cycle]	25,691,080
Total L1 Elapsed Cycles [cycle]	25,691,080	Total L2 Elapsed Cycles [cycle]	30,048,064
Total SMSP Elapsed Cycles [cycle]	102,764,320	Total DRAM Elapsed Cycles [cycle]	43,871,232

Workload Distribution

	Average	Min	Max	Sum
SM Active Cycles	623,261.55	610,009	636,956	24,930,462
SMSP Active Cycles	577,681.41	563,388	592,389	92,429,026
L1 Active Cycles	623,261.55	610,009	636,956	24,930,462
L2 Active Cycles	44,179.50	43,523	45,118	1,413,744
DRAM Active Cycles	62,735	62,320	63,484	501,880

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]	5,357,568	Branch Efficiency [%]	96.36
Branch Instructions Ratio [%]	0.08	Avg. Divergent Branches	1,024

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.