Result	Size	Time	Cycles	GPU	SM Frequency	Process	Attributes
Baseline 1 — 572 - jacobi_kernel	(32, 128, 1)x(32, 8, 1)	14.59 us	33,534	0 - NVIDIA GeForce RTX 5060 Ti	2.28 Ghz	[345743] jacobi2D_pointer_swap_1024.exe	

Summary | Details | Source | Context | Comments | Raw | Session

Compare | Tools | View | Export

GPU Speed Of Light Throughput

GPU Throughput Chart

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.

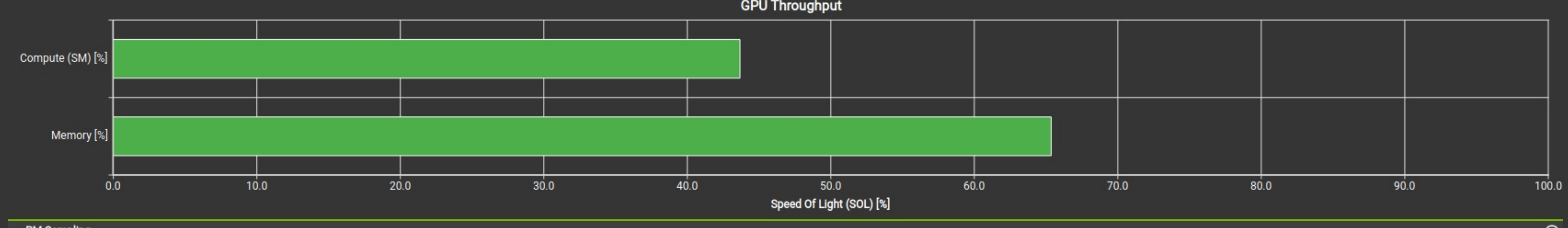
Compute (SM) Throughput [%]	43.67	Duration [us]	14.59
Memory Throughput [%]	65.36	Elapsed Cycles [cycle]	33,534
L1/TEX Cache Throughput [%]	48.26	SM Active Cycles [cycle]	30,130.19
L2 Cache Throughput [%]	45.86	SM Frequency [Ghz]	2.28
DRAM Throughput [%]	65.36	DRAM Frequency [Ghz]	13.75

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.

Key Performance Indicators

Roofline Analysis The ratio of peak float (FP32) to double (FP64) performance on this device is 64:1. The workload achieved 2% of this device's FP32 peak performance and 0% of its FP64 peak performance. See the Profiling Guide for more details on roofline analysis.

GPU Throughput

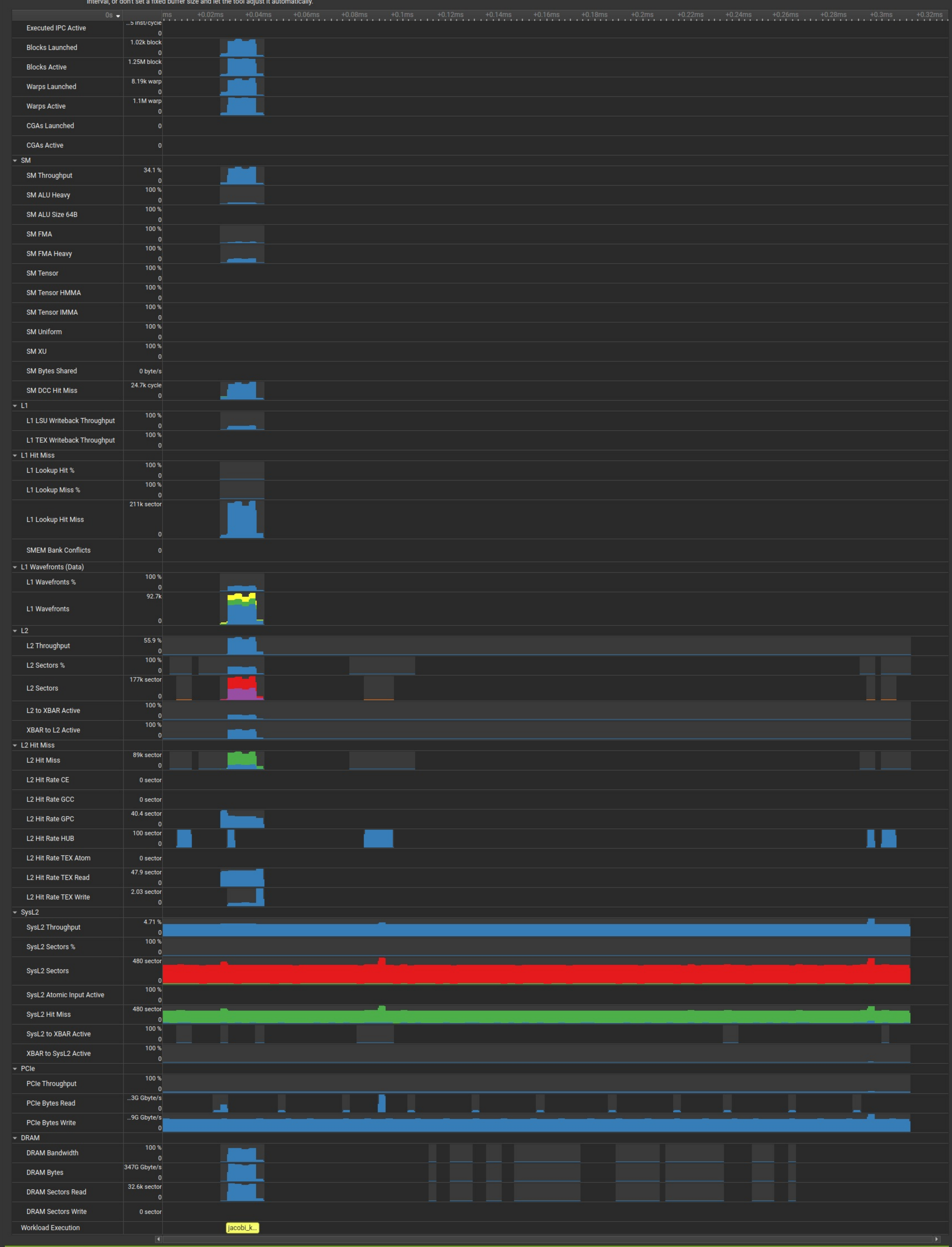


PM Sampling

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 [us]	3	# Pass Groups	1
Maximum Buffer Size [Mbyte]	8.26	-	-

PM Sampling Data Sampling interval is larger than 10% of the workload duration, which likely results in very few collected samples. For better results, use the --pm-sampling-interval option to reduce the sampling interval. Use --pm-sampling-buffer-size to increase the sampling buffer size for the smaller interval, or don't set a fixed buffer size and let the tool adjust it automatically.



Compute Workload Analysis

Pipe Utilization (Elapsed Cycles)

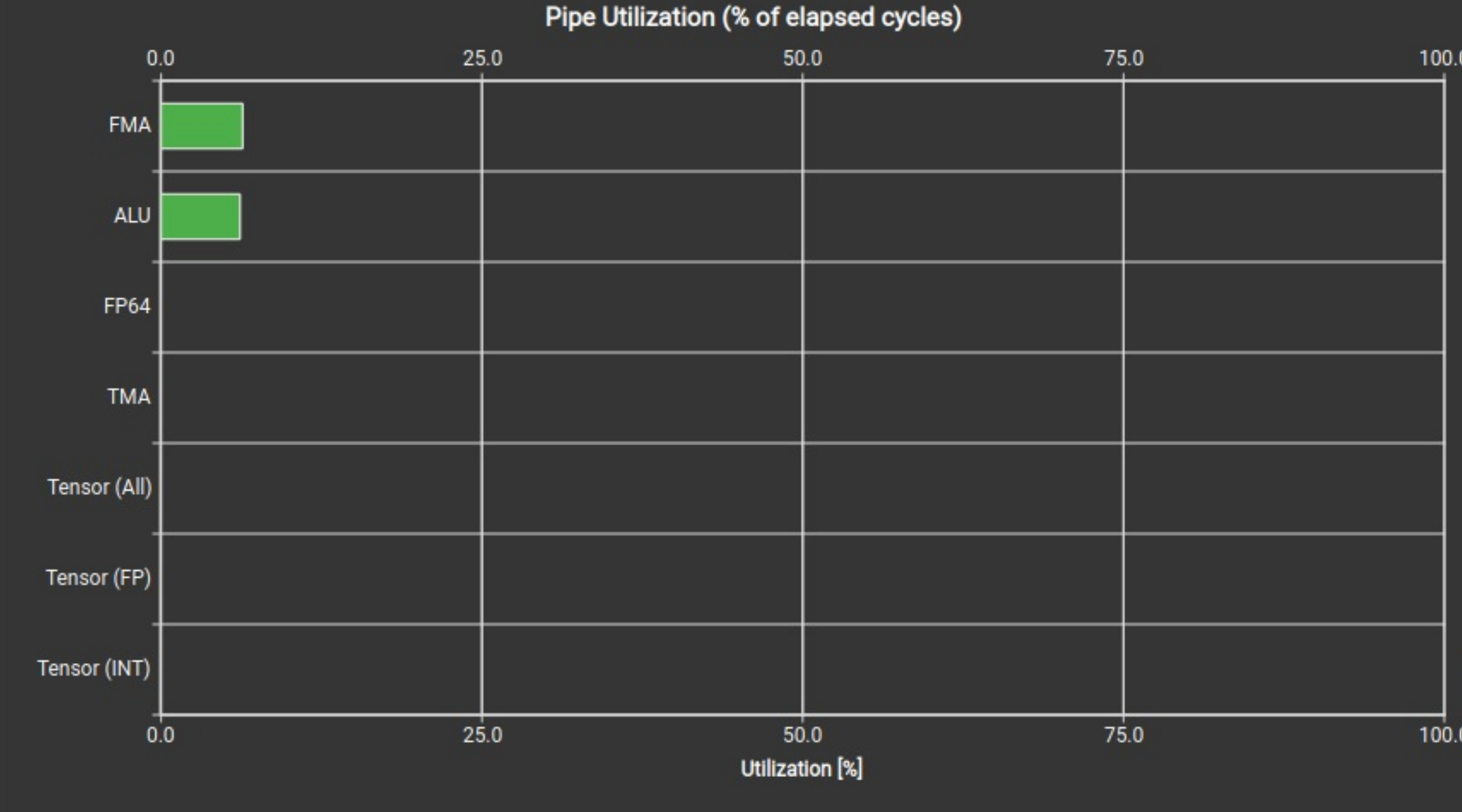
Detailed analysis of the compute resources of the streaming multiprocessors (SM), including the achieved instructions per clock (IPC) and the utilization of each available pipeline. Pipelines with very high utilization might limit the overall performance.

Executed Ipc Elapsed [inst/cycle]	1.12	SM Busy [%]	27.98
Executed Ipc Active [inst/cycle]	1.24	Issue Slots Busy [%]	27.98
Issued Ipc Active [inst/cycle]	1.24		

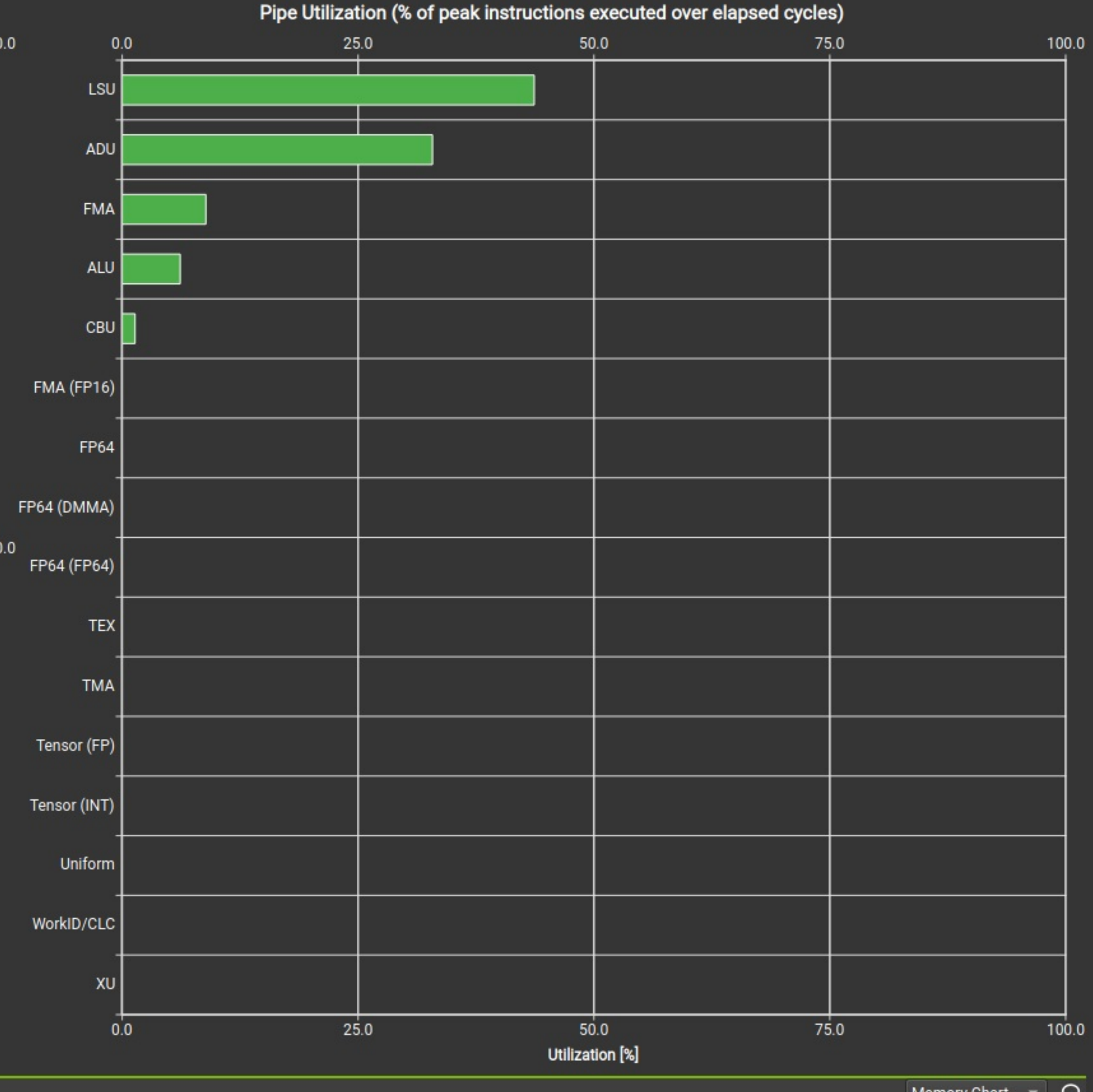
Low Utilization
Est. Local Speedup: 93.65% All compute pipelines are under-utilized. Either this workload is very small or it doesn't issue enough warps per scheduler. Check the Launch Statistics and Scheduler Statistics sections for further details.

Key Performance Indicators

Pipe Utilization (% of elapsed cycles)



Pipe Utilization (% of peak instructions executed over elapsed cycles)



Memory Workload Analysis

Memory Chart

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]	287.67	Mem Busy [%]	37.35
L1/TEX Hit Rate [%]	57.96	Max Bandwidth [%]	65.36
L2 Hit Rate [%]	26.72	Mem Pipes Busy [%]	43.67
L2 Compression Input Sectors [sector]	0	Local Memory Spilling Requests	0
L2 Compression Ratio	0	Local Memory Spilling Request Overhead [%]	0
L2 Compression Success Rate [%]	0	L2 Persisting Size [Mbyte]	6.29

L1TEX Global Load Access Pattern
Est. Speedup: 3.93% The memory access pattern for global loads from L1TEX might not be optimal. On average, only 29.1 of the 32 bytes transmitted per sector are utilized by each thread. This could possibly be caused by a stride between threads. Check the Source Counters section for uncoalesced global loads.

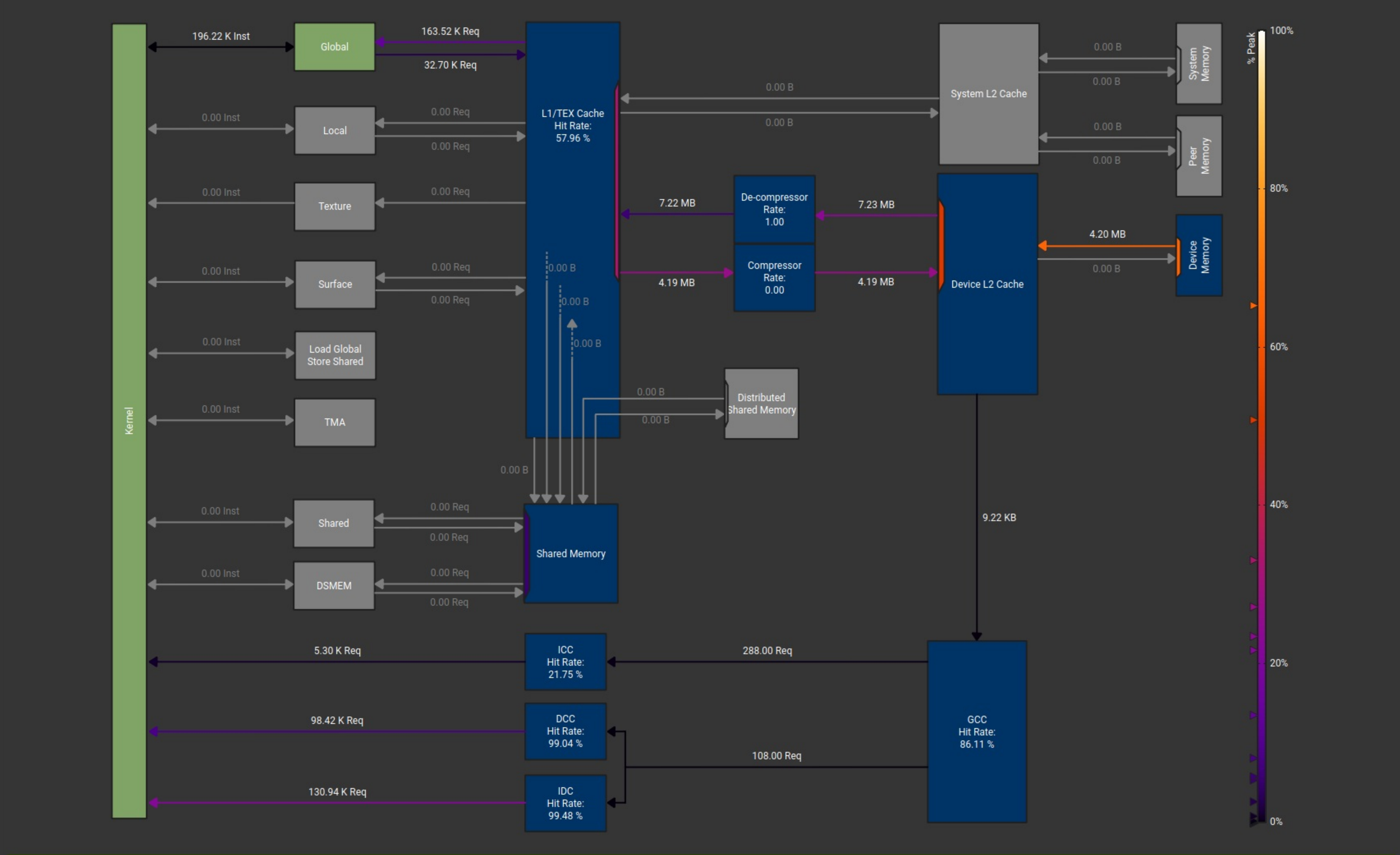
Key Performance Indicators

DRAM Global Store Access Pattern
Est. Speedup: 0.13% The memory access pattern for global stores to DRAM might not be optimal. On average, only 31.9 of the 32 bytes transmitted per sector are utilized by each thread. This applies to the 100.0% of sectors missed in L2. This could possibly be caused by a stride between threads. Check the Source Counters section for uncoalesced global stores.

Key Performance Indicators

Memory Chart

Values: Transfer Size Inactivity: Greyed Out



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]	8.61	No Eligible [%]	69.08
Eligible Warps Per Scheduler [warp]	0.48	One or More Eligible [%]	30.92
Issued Warp Per Scheduler	0.31		

Issue Slot Utilization
Est. Local Speedup: 34.64% Every scheduler is capable of issuing one instruction per cycle, but for this workload each scheduler only issues an instruction every 3.2 cycles. This might leave hardware resources underutilized and may lead to less optimal performance. Out of the maximum of 12 warps per scheduler, this workload allocates an average of 8.61 active warps per scheduler, but only an average of 0.48 warps were eligible per cycle. Eligible warps are the subset of active warps that are ready to issue their next instruction. Every cycle with no eligible warp results in no instruction being issued and the issue slot remains unused. To increase the number of eligible warps, reduce the time the active warps are stalled by inspecting the top stall reasons on the Warp State Statistics and Source Counters sections.

Key Performance Indicators

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]	27.86	Avg. Active Threads Per Warp	31.96
Warp Cycles Per Executed Instruction [cycle]	27.86	Avg. Not Predicated Off Threads Per Warp	31.18

Long Scoreboard Stalls
Est. Speedup: 34.64% On average, each warp of this workload spends 19.9 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 culprit. To reduce the number of cycles waiting on L1TEX data accesses verify the memory access patterns are optimal for the target architecture, attempt to increase cache hit rates by increasing data locality (coalescing), or by changing the cache configuration. Consider moving frequently used data to shared memory. This stall type represents about 71.4% of the total average of 27.9 cycles between issuing two instructions.

Key Performance Indicators

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

Instruction Statistics

Opcode Category Chart

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]	13,41,888	Avg. Executed Instructions Per Scheduler [inst]	9,318.67
Issued Instructions [inst]	13,41,888	Avg. Issued Instructions Per Scheduler [inst]	9,318.67

FP32 Non-Fused Instructions
Est. Speedup: 3.18% This kernel executes 0 fused and 163520 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).

Key Performance Indicators

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	4,096	Function Cache Configuration	CachePreferNone
Cluster Size	0	Preferred Cluster Size	0
Registers Per Thread [register/thread]	20	Cluster Scheduling Policy	PolicySpread
Static Shared Memory Per Block [byte/block]	0	Block Size	256
Dynamic Shared Memory Per Block [byte/block]	0	Threads [thread]	10,48,576
Driver Shared Memory Per Block [Kbyte/block]	1.02	Waves Per SM	18.96
Shared Memory Configuration Size [Kbyte]	16.38	Uses Green Context	0
Stack Size	1,024	# SMs [SM]	36
# TPCs	18	Enabled TPC IDs	all

Occupancy

% Occupancy Graphs

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]	10
Theoretical Active Warps per SM [warp]	48	Block Limit Shared Mem [block]	16
Achieved Occupancy [%]	73.60	Block Limit Warps [block]	6
Achieved Active Warps Per SM [warp]	35.33	Block Limit SM [block]	24
Cluster Occupancy [%]	0	Block Limit Barriers [block]	24
Max Active Clusters [cluster]	0	Max Cluster Size [block]	8
Overall GPU Occupancy [%]	0		

Achieved Occupancy
Est. Speedup: 26.40% The difference between calculated theoretical (100.0%) and measured achieved occupancy (73.6%) 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.

Key Performance Indicators

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]	30,130.19	Average L1 Active Cycles [cycle]	30,130.19
Average L2 Active Cycles [cycle]	26,431.62	Average SMSP Active Cycles [cycle]	30,137.28
Average DRAM Active Cycles [cycle]	1,31,176	Total SM Elapsed Cycles [cycle]	11,98,764
Total L1 Elapsed Cycles [cycle]	11,98,764	Total L2 Elapsed Cycles [cycle]	4,78,880
Total SMSP Elapsed Cycles [cycle]	47,95,056	Total DRAM Elapsed Cycles [cycle]	8,02,816

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]	65,472	Branch Efficiency [%]	0
Branch Instructions Ratio [%]	0.05	Avg. Divergent Branches [branches]	0

Uncoalesced Global Accesses
Est. Speedup: 6.60% This kernel has uncoalesced global accesses resulting in a total of 63364 excessive sectors (7% of the total 848260 sectors). Check the L2 Theoretical Sectors Global Excessive table for the primary source locations. The CUDA Programming Guide has additional information on reducing uncoalesced device memory accesses.

Key Performance Indicators

L2 Theoretical Sectors Global Excessive

Location	Value	Value (%)
0x7a08077676e0 in jacobi_kernel	31,682	50
0x7a0807767690 in jacobi_kernel	31,682	50
0x7a0807767770 in jacobi_kernel	0	0
0x7a0807767700 in jacobi_kernel	0	0
0x7a08077676f0 in jacobi_kernel	0	0

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.