	Result	Size	Time	Cycles	GPU	SM Frequency	Process	Attributes
Current	S25 - convolutionSharedKernel	(2, 2, 1024)x(8, 8, 1)	23.90 us	26.102	0 - NVIDIA A100-SXM4-40GB	1.09 Ghz	[3876227] cnn_example	

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

GPU Speed Of Light Throughput

GPU Throughput Rooflines

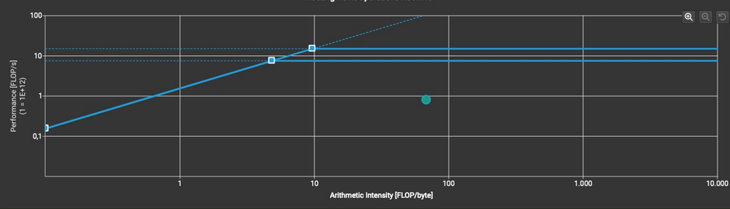
High-level overview of the throughput for compute and memory resources of the GPU. For each unit, the throughput reports the achieved percentage of utilization with respect to the theoretical maximum. Breakdowns show the throughput for each individual sub-metric of Compute and Memory to clearly identify 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 [%]	61.85	Duration [us]	23.90
Memory Throughput [%]	64.41	Elapsed Cycles [cycle]	26102
L1/TEX Cache Throughput [%]	73.47	SM Active Cycles [cycle]	22814.86
L2 Cache Throughput [%]	25.38	SM Frequency [Ghz]	1.09
DRAM Throughput [%]	0.74	DRAM Frequency [Ghz]	1.21

Balanced Throughput Compute and Memory are well-balanced: To reduce runtime, both computation and memory traffic must be reduced. Check both the Compute Workload Analysis and Memory Workload Analysis sections.

Roofline Analysis The ratio of peak float (fp32) to double (fp64) performance on this device is 2:1. The kernel achieved 5% 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.

Floating Point Operations Roofline



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]	20000	# Pass Groups	4
Maximum Buffer Size [Mbyte]	1.05	Dropped Samples [sample]	9

PM Sampling Data Sampling interval is larger than 10% of the workload duration, which likely results in very few collected samples.

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]	2.46	SM Busy [%]	70.55
Executed Ipc Active [inst/cycle]	2.81	Issue Slots Busy [%]	70.55
Issued Ipc Active [inst/cycle]	2.82		

Balanced FMA is the highest-utilized pipeline (51.4%) based on active cycles, taking into account the rates of its different instructions. It executes 32-bit floating point (FADD, FMUL, FMAD, ...) and integer (IMUL, IMAD) operations. It is well-utilized, but should not be a bottleneck.

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.

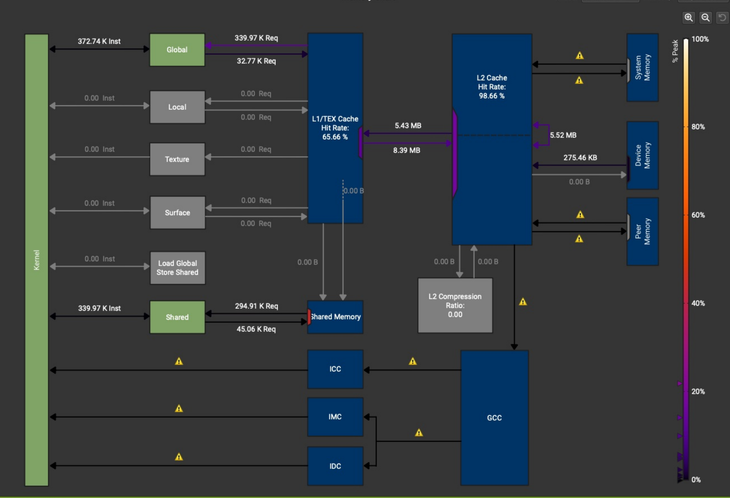
Memory Throughput [Gbyte/s]	11.52	Mem Busy [%]	64.41
L1/TEX Hit Rate [%]	65.66	Max Bandwidth [%]	25.94
L2 Hit Rate [%]	98.66	Mem Pipes Busy [%]	25.94
L2 Compression Success Rate [%]	0	L2 Compression Ratio	0

L1TEX Global Load Access Pattern Est. Speedup: 43.06%
The memory access pattern for global loads from L1TEX might not be optimal. On average, only 10.6 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

L1TEX Global Store Access Pattern Est. Speedup: 32.20%
The memory access pattern for global stores to L1TEX might not be optimal. On average, only 16.0 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 stores.
Key Performance Indicators

Shared Load Bank Conflicts Est. Speedup: 54.97%
The memory access pattern for shared loads might not be optimal and causes on average a 4.0 - way bank conflict across all 294912 shared load requests.This results in 884736 bank conflicts, which represent 74.82% of the overall 1182494 wavefronts for shared loads. Check the Source Counters section for uncoalesced shared loads.
Key Performance Indicators

Memory Chart



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]	8.95	No Eligible [%]	28.10
Eligible Warps Per Scheduler [warp]	2.62	One or More Eligible [%]	71.90
Issued Warp Per Scheduler	0.72		

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 scheduler's 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]	12.45	Avg. Active Threads Per Warp	31.45
Warp Cycles Per Executed Instruction [cycle]	12.51	Avg. Not Predicated Off Threads Per Warp	28.33

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]	6918144	Avg. Executed Instructions Per Scheduler [inst]	16014.22
Issued Instructions [inst]	6953854	Avg. Issued Instructions Per Scheduler [inst]	16096.88

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	4096	Function Cache Configuration	CachePreferNone
Registers Per Thread [register/thread]	40	Static Shared Memory Per Block [byte/block]	0
Block Size	64	Dynamic Shared Memory Per Block [Kbyte/block]	1.30
Threads [thread]	262144	Driver Shared Memory Per Block [Kbyte/block]	1.02
Waves Per SM	1.58	Shared Memory Configuration Size [Kbyte]	102.40
Uses Green Context	0	# SMs [SM]	108

Tail Effect Est. Speedup: 50.00%
A wave of thread blocks is defined as the maximum number of blocks that can be executed in parallel on the target GPU. The number of blocks in a wave depends on the number of multiprocessors and the theoretical occupancy of the kernel. This kernel launch results in 1 full waves and a partial wave of 1503 thread blocks. Under the assumption of a uniform execution duration of all thread blocks, the partial wave may account for up to 50.0% of the total kernel runtime with a lower occupancy of 26.7%. Try launching a grid with no partial wave. The overall impact of this tail effect also lessens with the number of full waves executed for a grid. See the Hardware Model description for more details on launch configurations.
Key Performance Indicators

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 [%]	75	Block Limit Registers [block]	24
Theoretical Active Warps per SM [warp]	48	Block Limit Shared Mem [block]	42
Achieved Occupancy [%]	54.98	Block Limit Warps [block]	32
Achieved Active Warps Per SM [warp]	35.19	Block Limit SM [block]	32

Achieved Occupancy Est. Local Speedup: 26.70%
The difference between calculated theoretical (75.0%) and measured achieved occupancy (55.0%) 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

Theoretical Occupancy Est. Local Speedup: 25.00%
The 12.00 theoretical warps per scheduler this kernel can issue according to its occupancy are below the hardware maximum of 16. This kernel's theoretical occupancy (75.0%) is limited by the number of required registers.
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]	22814.86	Average L1 Active Cycles [cycle]	22814.86
Average L2 Active Cycles [cycle]	20263.08	Average SMSP Active Cycles [cycle]	22386.67
Average DRAM Active Cycles [cycle]	215.20	Total SM Elapsed Cycles [cycle]	2810794
Total L1 Elapsed Cycles [cycle]	2810794	Total L2 Elapsed Cycles [cycle]	2000960
Total SMSP Elapsed Cycles [cycle]	11243176	Total DRAM Elapsed Cycles [cycle]	1160192

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 scheduler's fail to issue every cycle.

Branch Instructions [inst]	696320	Branch Efficiency [%]	99.35
Branch Instructions Ratio [%]	0.10	Avg. Divergent Branches [branches]	9.48

Uncoalesced Global Accesses Est. Speedup: 22.74%
This kernel has uncoalesced global accesses resulting in a total of 229376 excessive sectors (28% of the total 817152 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 (%)
0x7ffe81455d20 in convolutionSharedKernel	131.072	57
0x7ffe81454a60 in convolutionSharedKernel	24.576	11
0x7ffe81454960 in convolutionSharedKernel	24.576	11
0x7ffe81454a50 in convolutionSharedKernel	20.480	9
0x7ffe81453db0 in convolutionSharedKernel	16.384	7

Uncoalesced Shared Accesses Est. Speedup: 63.33%
This kernel has uncoalesced shared accesses resulting in a total of 884736 excessive wavefronts (72% of the total 1224704 wavefronts). Check the L1 Wavefronts Shared Excessive table for the primary source locations. The CUDA Best Practices Guide has an example on optimizing shared memory accesses.
Key Performance Indicators

L1 Wavefronts Shared Excessive

Location	Value	Value (%)
0x7ffe81455ca0 in convolutionSharedKernel	294.912	33
0x7ffe81455c70 in convolutionSharedKernel	294.912	33
0x7ffe81455c20 in convolutionSharedKernel	294.912	33
0x7ffe81455b00 in convolutionSharedKernel	0	0
0x7ffe81455a90 in convolutionSharedKernel	0	0