

...ant to see detailed metrics. Double click on

Compute Throughput	Memory Throughput #
70.60	84.18

62.9 for naive

94.4 for naive

Compute and memory to clearly identify the highest contributor. High-level overview of the utilization for compute and memory resource

Compute (SM) Throughput [%]	70.60
Memory Throughput [%]	84.18
L1/TEX Cache Throughput [%]	88.80
L2 Cache Throughput [%]	8.95
DRAM Throughput [%]	0.78

Compute Throughput Breakdown		Sp
SM: Inst Executed Pipe Lsu [%]	70.60	→
SM: Mio2rf Writeback Active [%]	40.13	
SM: Issue Active [%]	39.28	
SM: Inst Executed [%]	39.25	
SM: Mio Inst Issued [%]	36.95	
SM: Pipe Fma Cycles Active [%]	26.54	
SM: Pipe Alu Cycles Active [%]	12.43	
SM: Inst Executed Pipe Adu [%]	6.60	
SM: Mio Pq Read Cycles Active [%]	3.25	
SM: Mio Pq Write Cycles Active [%]	3.25	
SM: Pipe Tensor Cycles Active [%]	0.86	
SM: Pipe Shared Cycles Active [%]	0.86	
SM: Inst Executed Pipe Uniform [%]	0.85	
SM: Inst Executed Pipe Cbu Pred On Any [%]	0.05	
SM: Inst Executed Pipe lpa [%]	0	
SM: Inst Executed Pipe Fp16 [%]	0	
SM: Inst Executed Pipe Tex [%]	0	
SM: Pipe Fp64 Cycles Active [%]	0	
SM: Inst Executed Pipe Xu [%]	0	
IDC: Request Cycles Active [%]	0	

up from 63 in naive

(due to SWEET orchestration) (L2 we're using SWEET now)

up from 32% in naive

(lots of register file writes/reads)

down from 31% in naive (likely due to

singlekey L2S
Storage on L2

lack of vectorized
loads)

L1 hit rate

88 \Rightarrow 1.7 b/c of

testing

Mem Busy [%]	84.18
Max Bandwidth [%]	60.15
Mem Pipes Busy [%]	70.60
L2 Compression Ratio	0

down b/c less strain
on L1

up b/c
more smem +
4s units constantly
working

very low
bank conflicts

Shared Memory					
	Instructions	Requests	Wavefronts	% Peak	Bank Conflicts
Shared Load	41,943,040	41,943,040	50,365,622	76.90	12,418
Shared Load Matrix	0	0	0	0	0
Shared Store	2,097,152	2,097,152	2,363,830	3.61	264,385
Shared Store From Global Load	0	0	0	0	4,218
Shared Atomic	0	0	0	0	0
Other	-	-	135,167	0.21	0
Total	44,040,192	44,040,192	52,864,619	80.72	281,021

very good
arithmetic intensity
Store 1x, Load 10x

very good
smem usage
for smem \rightarrow regs

L2 Cache									
Requests	Sectors	Sectors/Req	% Peak	Hit Rate	Bytes	Throughput	Sector Miss		
L1/TEX Store	32,248	131,072	4	0.19	100	4,194,304	78,841,405,854		
L1/TEX Atomic ALU	0	0	0	0	0	0	0		
L1/TEX Atomic CAS	0	0	0	0	0	0	0		
L1/TEX Reduction	0	0	0	0	0	0	0		
L1/TEX Total	2,096,568	8,381,232	4.00	9.08	94.48	28,827,984	597,423,417,604.36		
L1/TEX Hit	0	0	0	0	0	0	0		
L1/TEX Miss	0	0	0	0	0	0	0		
L2 Fabric Total	96,134	344,308	3.51	0.75	92.78	11,077,886	20,841,888,619.85		
GPU Total	2,202,223	8,725,540	3.97	6.31	94.46	29,935,870	528,465,222,602.29		

L2 Cache Eviction Policies									
First	Hit Rate	Last	Hit Rate	Normal	Normal Demote	Hit Rate			
L1/TEX Load	0	0	0	0	0	94.36			
L1/TEX Store	0	0	0	0	0	100			
L1/TEX Atomic	0	0	0	0	0	0			
L1/TEX Total	0	0	0	0	0	94.87			
L1/TEX Hit	0	0	0	0	0	94.87			
L1/TEX Miss	0	0	0	0	0	0			
GPU Total	2,565	100	0	0	0	94.00			

Device Memory									
Sectors	% Peak	Bytes	Throughput	Hit Rate					
262,148	0	8,388,726	9,951,078,708.83	0					
262,148	0.48	8,388,726	9,951,078,708.83	0					

in tiled

Stream is now fairly on the L2

responsibility

tiled is faster and uses fewer reports

L2 Cache									
Requests	Sectors	Sectors/Req	% Peak	Hit Rate	Bytes	Throughput	Sector Miss		
L1/TEX Store	65,536	131,072	2	0.12	100	4,194,304	48,934,466,965.57		
L1/TEX Atomic ALU	0	0	0	0	0	0	0		
L1/TEX Atomic CAS	0	0	0	0	0	0	0		
L1/TEX Reduction	0	0	0	0	0	0	0		
L1/TEX Total	1,197,825	16,226,719	1.35	10.76	97.29	51,925,008	603,562,762,817.01		
L1/TEX Hit	0	0	0	0	0	0	0		
L1/TEX Miss	0	0	0	0	0	0	0		
L2 Fabric Total	187,780	381,840	1.87	6.47	72.38	17,253,760	13,081,346,698.41		
GPU Total	12,192,303	16,589,549	1.86	7.33	96.86	530,865,568	617,078,994,509.75		

L2 Cache Eviction Policies									
First	Hit Rate	Last	Hit Rate	Normal	Normal Demote	Hit Rate			
L1/TEX Load	0	0	0	0	0	97.77			
L1/TEX Store	0	0	0	0	0	100			
L1/TEX Atomic	0	0	0	0	0	0			
L1/TEX Total	0	0	0	0	0	97.88			
L1/TEX Hit	0	0	0	0	0	97.88			
L1/TEX Miss	0	0	0	0	0	0			
GPU Total	3,549	100	0	0	0	96.86			

Device Memory									
Sectors	% Peak	Bytes	Throughput	Hit Rate					
262,148	0	8,388,726	9,951,078,708.83	0					
262,148	0.48	8,388,726	9,951,078,708.83	0					

In aive L2 saved me

faster

No. Digible ↓ from 65 naïve to

(dose to latency hiding)

39 in filed

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. States 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 binary and user code, these metrics show the combined values.

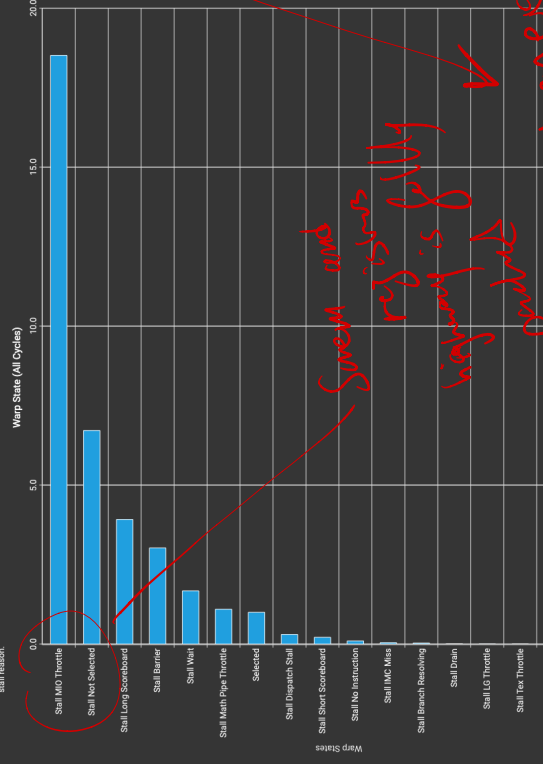
Warp Cycles Per Issued Instruction (CPI) 36.66 Avg. Active Threads Per Warp 31.97

On average, each warp of this kernel spends 18.5 cycles being stalled waiting for the MIO (memory input/output) instruction queue to be not full. This stall reason is high in cases of extreme utilization of the MIO pipelines, which include special math instructions, dynamic branches, as well as shared memory instructions. When caused by shared memory instructions, trying to use fewer data wider banks can reduce pipeline pressure. This stall type represents about 50.3% of the total average of 36.6 cycles between issuing two instructions.

Key Performance Indicators

Metric Name	Value	Guidance
emp_active_avg_per_cycle_active	14.654	15.15 is 8.6
emp_active_avg_per_cycle_stalled_mio	18.161	18.51 is 50.3

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



Stream and registers fully bypassed is fully generated

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. States 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 binary and user code, these metrics show the combined values.

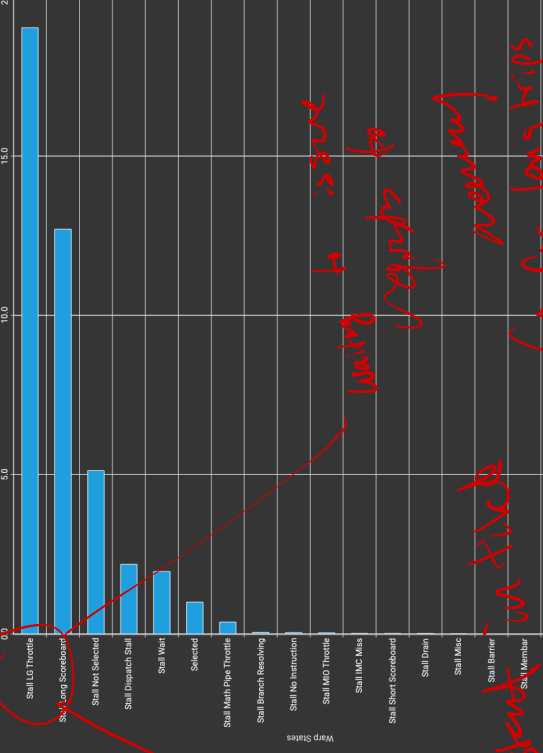
Warp Cycles Per Issued Instruction (CPI) 42.56 Avg. Active Threads Per Warp 31.97

On average, each warp of this kernel spends 11.5 cycles being stalled waiting for the L2 (local store queue for local and global L2) memory operations to be not full. Typically, this is caused by excessive use of the L2 memory. If the kernel has excessive register pressure causing by spills, it is applicable, consider combining multiple lower width registers into fewer wider memory operations and try interleaving memory operations and math instructions. This stall type represents about 41.2% of the total average of 42.56 cycles between issuing two instructions.

Key Performance Indicators

Metric Name	Value	Guidance
emp_active_avg_per_cycle_active	14.654	15.15 is 8.6
emp_active_avg_per_cycle_stalled_mio	18.161	18.51 is 50.3

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



waiting to issue reports to memory

more expensive - we fixed b/c of binary stream latency (A for long trips was full)

Mediatek WinBox X 3100-MediatekRecrpt

Summary Details Source Context Comments Raw Session

Current 519-matmul_d1d_kernel S19-matmul_d1d_kernel GPU 0-NVIDIA A100-SXM4-80GB 1.15 GHz SM Frequency Process Attributes

Cycles 608,909 Time 528.64 us Size (02:32, 1K(02:32:1)) 528.64 us [3213] bench_a100

Left View: SAS - Right View: None - Navigating By: Instructions Executed

Source: matmul_d1d_kernel

# Label	Address	Scoreboard	Registers	Live	Instruction Category	Instructions Executed	Attributed Stalls	Warp Stal Sampling (All Cycles)
33	000977ab 85451369		18	18	Integer	1,028	0	0.18%
34	000977ab 8545136a		19	19	Movement	1,028	0	0.18%
35	000977ab 8545136b		19	19	Movement	1,028	0	0.18%
36	000977ab 8545136c		19	19	Movement	1,028	0	0.18%
37	000977ab 8545136d		19	19	Movement	1,028	0	0.18%
38	000977ab 8545136e		19	19	Integer	1,028	0	0.18%
39	000977ab 8545136f		19	19	Integer	1,028	0	0.18%
40	000977ab 85451370		19	19	Integer	1,028	0	0.18%
41	000977ab 85451371		19	19	Integer	1,028	0	0.18%
42	000977ab 85451372		19	19	Integer	1,028	0	0.18%
43	000977ab 85451373		19	19	Integer	1,028	0	0.18%
44	000977ab 85451374		19	19	Integer	1,028	0	0.18%
45	000977ab 85451375		19	19	Integer	1,028	0	0.18%
46	000977ab 85451376		19	19	Integer	1,028	0	0.18%
47	000977ab 85451377		19	19	Integer	1,028	0	0.18%

✓
leez