Analysis Report

PointInPolyhedron0(float3*, float3*, char*, unsigned int, unsigned int)

Duration	7.424 µs
Grid Size	[100,1,1]
Block Size	[1024,1,1]
Registers/Thread	32
Shared Memory/Block	4 B
Shared Memory Requested	96 KiB
Shared Memory Executed	96 KiB
Shared Memory Bank Size	4 B

[0] GeForce GTX 1080

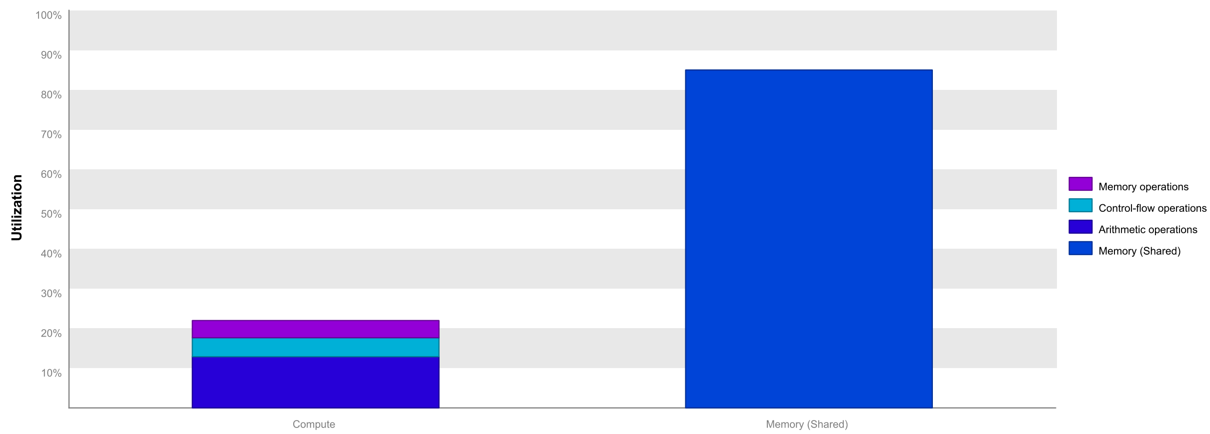
GPU UUID	GPU-41d07ef5-05a7-c37d-84ad-0247909edad3
Compute Capability	6.1
Max. Threads per Block	1024
Max. Threads per Multiprocessor	2048
Max. Shared Memory per Block	48 KiB
Max. Shared Memory per Multiprocessor	96 KiB
Max. Registers per Block	65536
Max. Registers per Multiprocessor	65536
Max. Grid Dimensions	[2147483647, 65535, 65535]
Max. Block Dimensions	[1024, 1024, 64]
Max. Warps per Multiprocessor	64
Max. Blocks per Multiprocessor	32
Half Precision FLOP/s	69.34 GigaFLOP/s
Single Precision FLOP/s	8.876 TeraFLOP/s
Double Precision FLOP/s	277.36 GigaFLOP/s
Number of Multiprocessors	20
Multiprocessor Clock Rate	1.734 GHz
Concurrent Kernel	true
Max IPC	6
Threads per Warp	32
Global Memory Bandwidth	320.32 GB/s
Global Memory Size	8 GiB
Constant Memory Size	64 KiB
L2 Cache Size	2 MiB
Memcpy Engines	2
PCIe Generation	3
PCIe Link Rate	8 Gbit/s
PCIe Link Width	16

1. Compute, Bandwidth, or Latency Bound

The first step in analyzing an individual kernel is to determine if the performance of the kernel is bounded by computation, memory bandwidth, or instruction/memory latency. The results below indicate that the performance of kernel "PointInPolyhedron0" is most likely limited by memory bandwidth. You should first examine the information in the "Memory Bandwidth" section to determine how it is limiting performance.

1.1. Kernel Performance Is Bound By Memory Bandwidth

For device "GeForce GTX 1080" the kernel's compute utilization is significantly lower than its memory utilization. These utilization levels indicate that the performance of the kernel is most likely being limited by the memory system. For this kernel the limiting factor in the memory system is the bandwidth of the Shared memory.



2. Memory Bandwidth

Memory bandwidth limits the performance of a kernel when one or more memories in the GPU cannot provide data at the rate requested by the kernel. The results below indicate that the kernel is limited by the bandwidth available to the shared memory.

2.1. GPU Utilization Is Limited By Memory Bandwidth

The following table shows the memory bandwidth used by this kernel for the various types of memory on the device. The table also shows the utilization of each memory type relative to the maximum throughput supported by the memory. The results show that the kernel's performance is potentially limited by the bandwidth available from one or more of the memories on the device.

Optimization: Try the following optimizations for the memory with high bandwidth utilization.
Shared Memory - If possible use 64-bit accesses to shared memory and 8-byte bank mode to achieved 2x throughput.
L2 Cache - Align and block kernel data to maximize L2 cache efficiency.
Unified Cache - Reallocate texture data to shared or global memory. Resolve alignment and access pattern issues for global loads and stores.
Device Memory - Resolve alignment and access pattern issues for global loads and stores.
System Memory (via PCIe) - Make sure performance critical data is placed in device or shared memory.

Transactions	Bandwidth	Utilization	
Shared Memory			
Shared Loads	105700	1,989.647 GB/s	
Shared Stores	102500	1,929.412 GB/s	
Shared Total	208200	3,919.059 GB/s	Idle Low Medium High Max
L2 Cache			
Reads	210	988.235 MB/s	
Writes	113	531.765 MB/s	
Total	323	1.52 GB/s	Idle Low Medium High Max
Unified Cache			
Local Loads	0	0 B/s	
Local Stores	0	0 B/s	
Global Loads	0	0 B/s	
Global Stores	100	470.588 MB/s	
Texture Reads	0	0 B/s	
Unified Total	100	470.588 MB/s	Idle Low Medium High Max
Device Memory			
Reads	108	508.235 MB/s	
Writes	4	18.824 MB/s	
Total	112	527.059 MB/s	Idle Low Medium High Max
System Memory			
[PCIe configuration: Gen3 x16, 8 Gbit/s]			
Reads	0	0 B/s	Idle Low Medium High Max
Writes	5	23.529 MB/s	Idle Low Medium High Max

3. Instruction and Memory Latency

Instruction and memory latency limit the performance of a kernel when the GPU does not have enough work to keep busy. The performance of latency-limited kernels can often be improved by increasing occupancy. Occupancy is a measure of how many warps the kernel has active on the GPU, relative to the maximum number of warps supported by the GPU. Theoretical occupancy provides an upper bound while achieved occupancy indicates the kernel's actual occupancy.

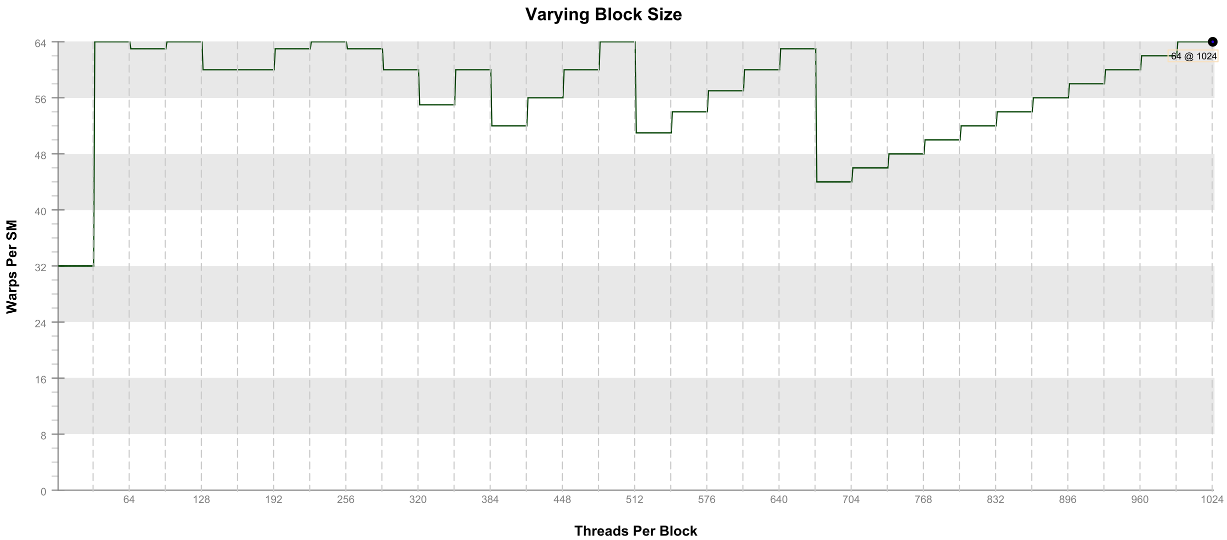
3.1. Occupancy Is Not Limiting Kernel Performance

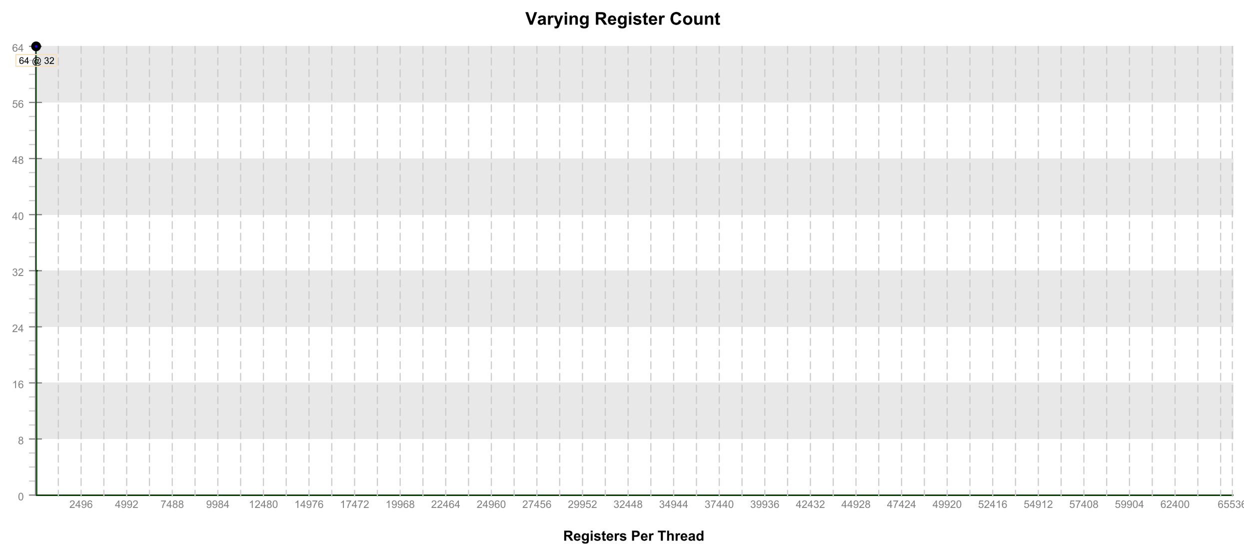
The kernel's block size, register usage, and shared memory usage allow it to fully utilize all warps on the GPU.

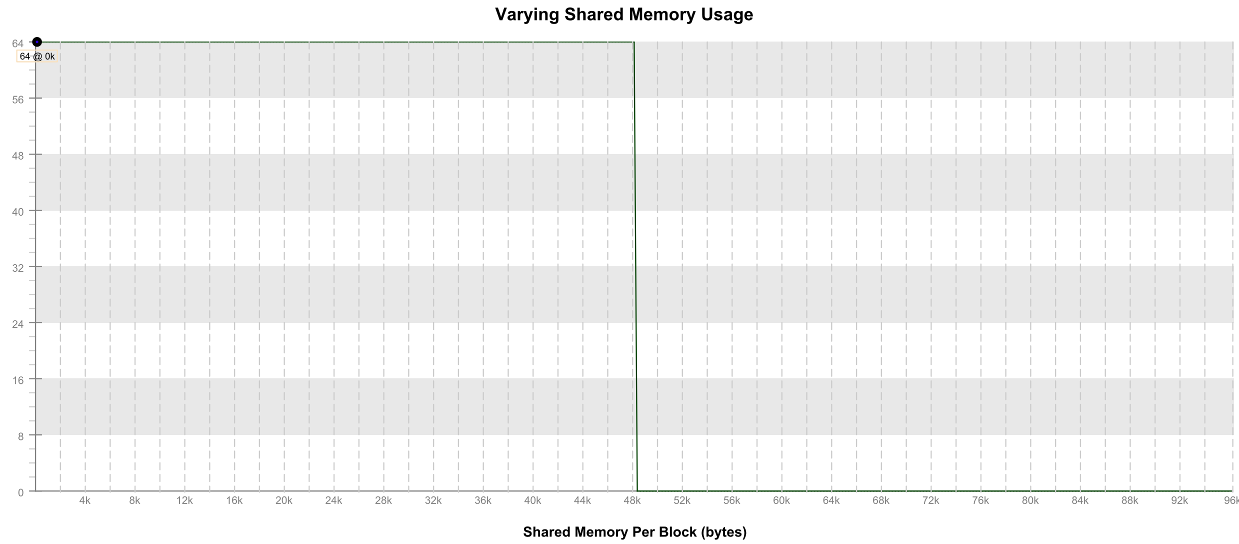
Variable	Achieved	Theoretical	Device Limit	Grid Size: [100,1,1] (100 blocks) Block Size: [1024,1,1] (1024 threads)
Occupancy Per SM				
Active Blocks		2	32	0 3 6 9 12 15 18 21 24 27 30 32
Active Warps	57.44	64	64	0 6 12 18 24 30 36 42 48 54 60 64
Active Threads		2048	2048	0 256 512 768 1024 1280 1536 1792 2048
Occupancy	89.7%	100%	100%	0% 25% 50% 75% 100%
Warps				
Threads/Block		1024	1024	0 128 256 384 512 640 768 896 1024
Warps/Block		32	32	0 3 6 9 12 15 18 21 24 27 30 32
Block Limit		2	32	0 3 6 9 12 15 18 21 24 27 30 32
Registers				
Registers/Thread		32	65536	0 8192 16384 24576 32768 40960 49152 57344 65536
Registers/Block		32768	65536	0 16k 32k 48k 64k
Block Limit		2	32	0 3 6 9 12 15 18 21 24 27 30 32
Shared Memory				
Shared Memory/Block		4	98304	0 16k 32k 48k 64k 80k 96k
Block Limit		384	32	0 3 6 9 12 15 18 21 24 27 30 32

3.2. Occupancy Charts

The following charts show how varying different components of the kernel will impact theoretical occupancy.


Varying Block Size
Warps Per SM
64
56
48
40
32
24
16
8
0
64
128
192
256
320
384
448
512
576
640
704
768
832
896
960
1024
64 @ 1024
Threads Per Block


Varying Register Count
64 @ 32
64
56
48
40
32
24
16
8
0
2496
4992
7488
9984
12480
14976
17472
19968
22464
24960
27456
29952
32448
34944
37440
39936
42432
44928
47424
49920
52416
54912
57408
59904
62400
65536
Registers Per Thread


Varying Shared Memory Usage
64 @ 0k
0
8
16
24
32
40
48
56
64
4k
8k
12k
16k
20k
24k
28k
32k
36k
40k
44k
48k
52k
56k
60k
64k
68k
72k
76k
80k
84k
88k
92k
96k
Shared Memory Per Block (bytes)

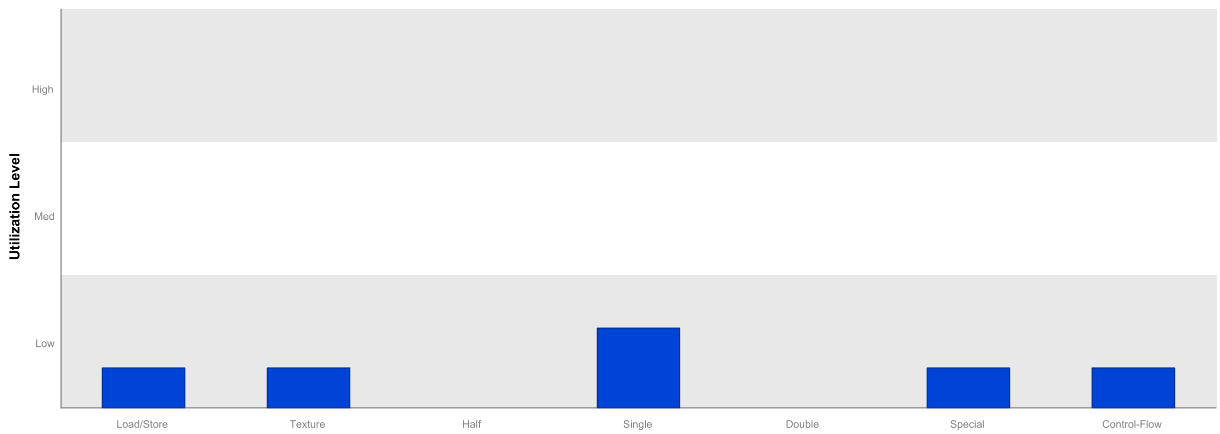
4. Compute Resources

GPU compute resources limit the performance of a kernel when those resources are insufficient or poorly utilized.

4.1. Function Unit Utilization

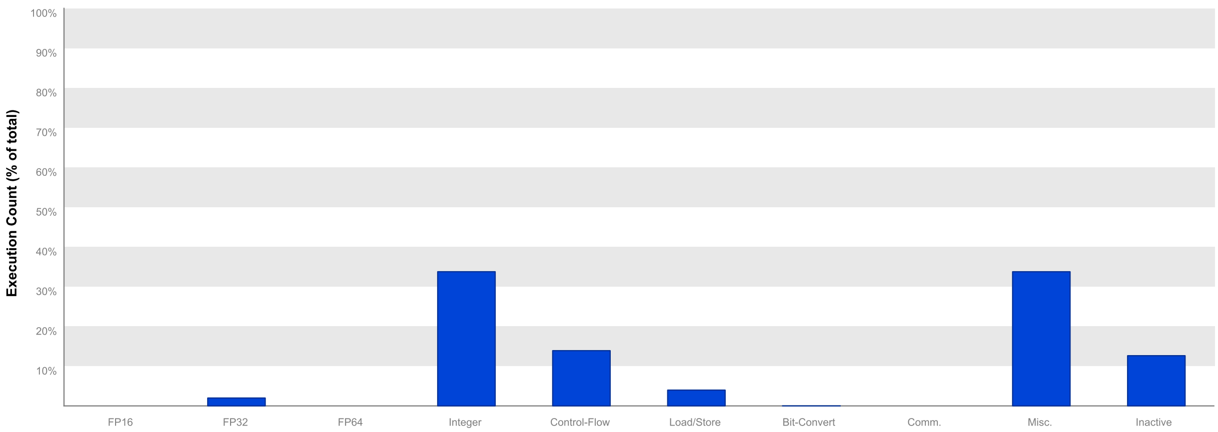
Different types of instructions are executed on different function units within each SM. Performance can be limited if a function unit is over-used by the instructions executed by the kernel. The following results show that the kernel's performance is not limited by overuse of any function unit.

Load/Store - Load and store instructions for shared and constant memory.
Texture - Load and store instructions for local, global, and texture memory.
Half - Half-precision floating-point arithmetic instructions.
Single - Single-precision integer and floating-point arithmetic instructions.
Double - Double-precision floating-point arithmetic instructions.
Special - Special arithmetic instructions such as sin, cos, popc, etc.
Control-Flow - Direct and indirect branches, jumps, and calls.



4.2. Instruction Execution Counts

The following chart shows the mix of instructions executed by the kernel. The instructions are grouped into classes and for each class the chart shows the percentage of thread execution cycles that were devoted to executing instructions in that class. The "Inactive" result shows the thread executions that did not execute any instruction because the thread was predicated or inactive due to divergence.



4.3. Floating-Point Operation Counts

The following chart shows the mix of floating-point operations executed by the kernel. The operations are grouped into classes and for each class the chart shows the percentage of thread execution cycles that were devoted to executing operations in that class. The results do not sum to 100% because non-floating-point operations executed by the kernel are not shown in this chart.