

Analysis Report

run_lbm(double[27]*, double[27]*, unsigned long, double, IndexBlock)

| | |
|-------------------------|---------------------------|
| Duration | 5.90161 ms (5,901,607 ns) |
| Grid Size | [30,1,1] |
| Block Size | [30,30,1] |
| Registers/Thread | 60 |
| Shared Memory/Block | 0 B |
| Shared Memory Requested | 48 KiB |
| Shared Memory Executed | 48 KiB |
| Shared Memory Bank Size | 4 B |

[0] GeForce GTX 765M

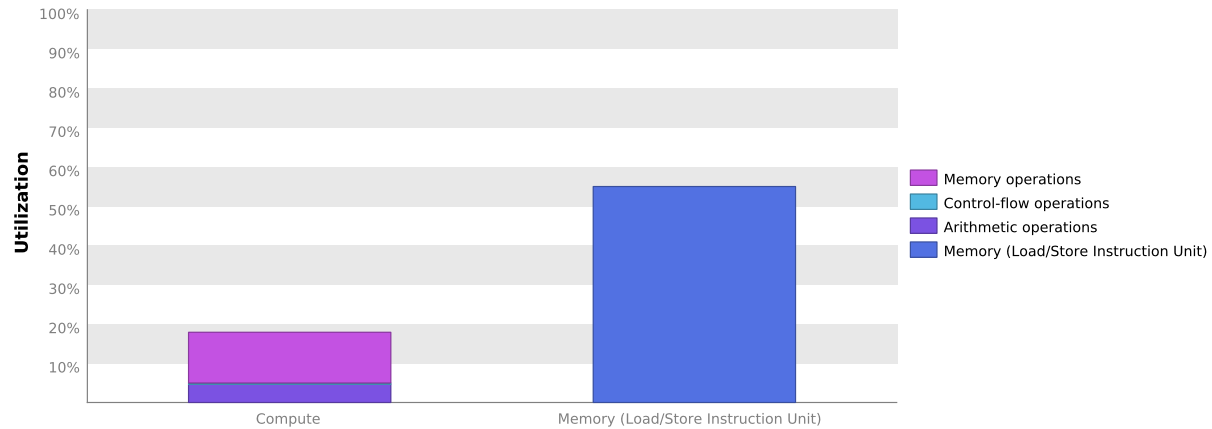
| | |
|---------------------------------------|--|
| GPU UUID | GPU-fe6c21ea-b7dc-9c6c-991c-33e89452d625 |
| Compute Capability | 3.0 |
| Max. Threads per Block | 1024 |
| Max. Threads per Multiprocessor | 2048 |
| Max. Shared Memory per Block | 48 KiB |
| Max. Shared Memory per Multiprocessor | 48 KiB |
| 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 | 16 |
| Single Precision FLOP/s | 1.325 TeraFLOP/s |
| Double Precision FLOP/s | 55.2 GigaFLOP/s |
| Number of Multiprocessors | 4 |
| Multiprocessor Clock Rate | 862.5 MHz |
| Concurrent Kernel | true |
| Max IPC | 7 |
| Threads per Warp | 32 |
| Global Memory Bandwidth | 64.128 GB/s |
| Global Memory Size | 1.952 GiB |
| Constant Memory Size | 64 KiB |
| L2 Cache Size | 256 KiB |
| Memcpy Engines | 1 |
| PCIe Generation | 2 |
| PCIe Link Rate | 5 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 "run_lbm" is most likely limited by instruction and memory latency. You should first examine the information in the "Instruction And Memory Latency" section to determine how it is limiting performance.

1.1. Kernel Performance Is Bound By Instruction And Memory Latency

This kernel exhibits low compute throughput and memory bandwidth utilization relative to the peak performance of "GeForce GTX 765M". These utilization levels indicate that the performance of the kernel is most likely limited by the latency of arithmetic or memory operations. Achieved compute throughput and/or memory bandwidth below 60% of peak typically indicates latency issues.



2. 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. The results below indicate that occupancy can be improved by reducing the number of registers used by the kernel.

2.1. GPU Utilization Is Limited By Register Usage

The kernel uses 60 registers for each thread (54000 registers for each block). This register usage is likely preventing the kernel from fully utilizing the GPU. Device "GeForce GTX 765M" provides up to 65536 registers for each block. Because the kernel uses 54000 registers for each block each SM is limited to simultaneously executing 1 block (29 warps). Chart "Varying Register Count" below shows how changing register usage will change the number of blocks that can execute on each SM.

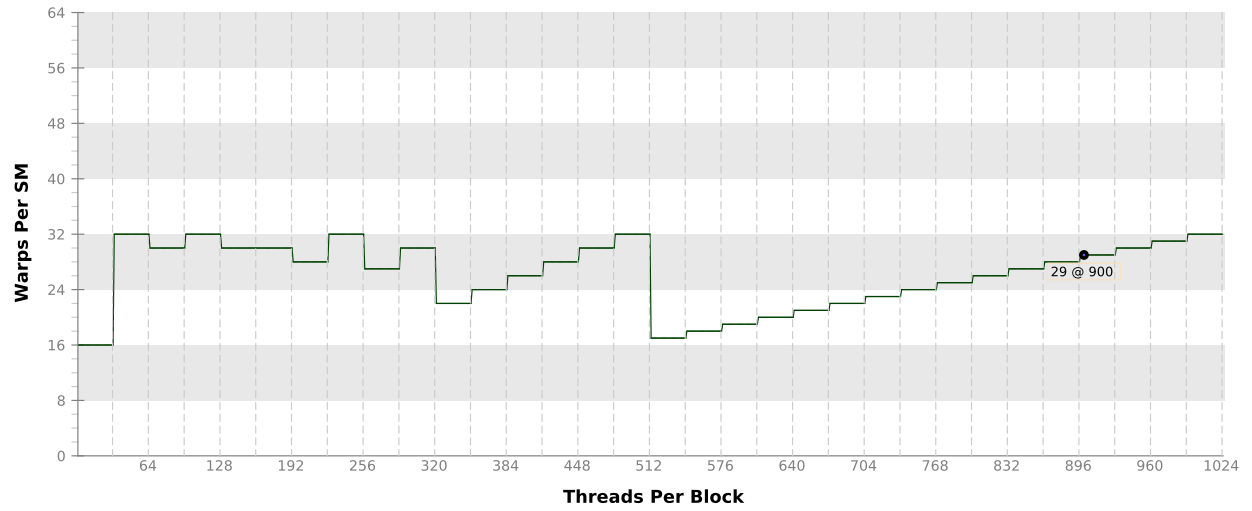
Optimization: Use the `-maxrregcount` flag or the `__launch_bounds__` qualifier to decrease the number of registers used by each thread. This will increase the number of blocks that can execute on each SM. On devices with Compute Capability 5.2 turning global cache off can increase the occupancy limited by register usage.

| Variable | Achieved | Theoretical | Device Limit | Grid Size: [30,1,1] (30 blocks) Block Size: [30,30,1] (900 threads) |
|---------------------|----------|-------------|--------------|---|
| Occupancy Per SM | | | | |
| Active Blocks | | 1 | 16 | |
| Active Warps | 27.38 | 29 | 64 | |
| Active Threads | | 928 | 2048 | |
| Occupancy | 42.8% | 45.3% | 100% | |
| Warps | | | | |
| Threads/Block | | 900 | 1024 | |
| Warps/Block | | 29 | 32 | |
| Block Limit | | 2 | 16 | |
| Registers | | | | |
| Registers/Thread | | 60 | 65536 | |
| Registers/Block | | 59392 | 65536 | |
| Block Limit | | 1 | 16 | |
| Shared Memory | | | | |
| Shared Memory/Block | | 0 | 49152 | |
| Block Limit | | 0 | 16 | |

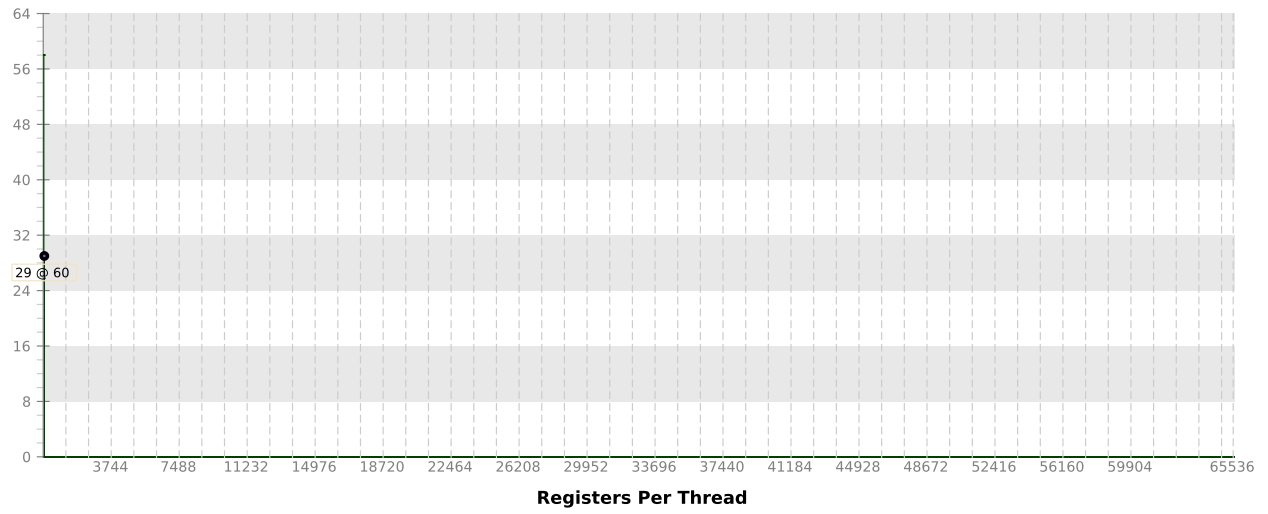
2.2. Occupancy Charts

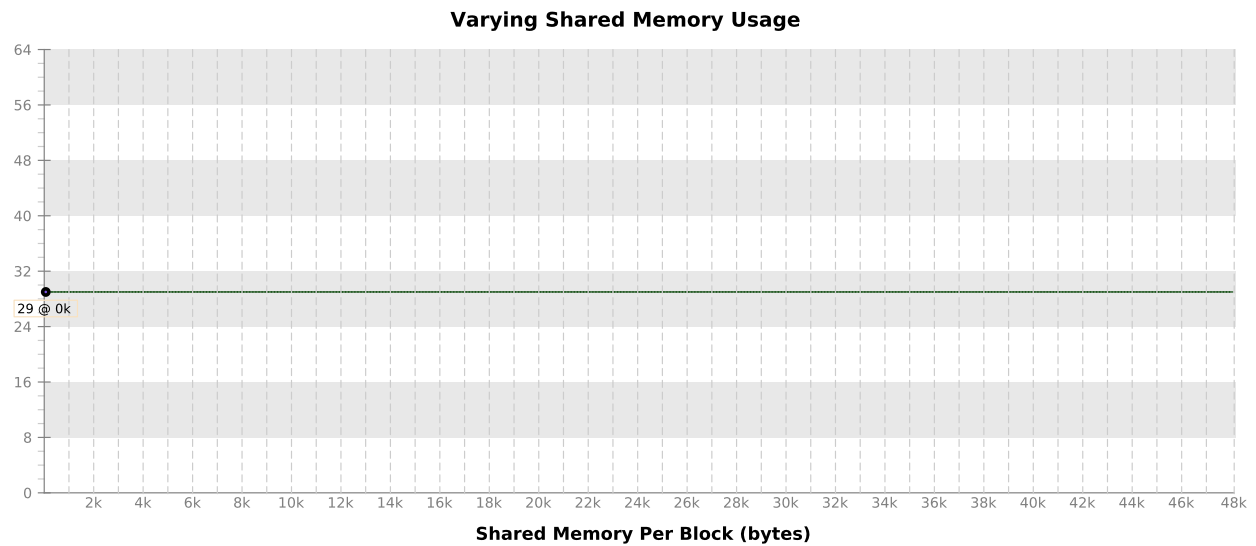
The following charts show how varying different components of the kernel will impact theoretical occupancy.

Varying Block Size



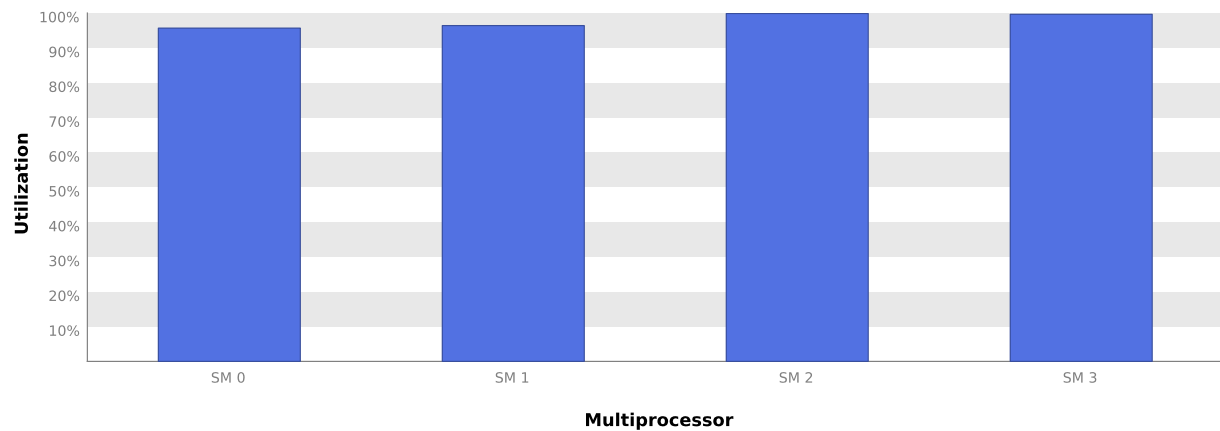
Varying Register Count





2.3. Multiprocessor Utilization

The kernel's blocks are distributed across the GPU's multiprocessors for execution. Depending on the number of blocks and the execution duration of each block some multiprocessors may be more highly utilized than others during execution of the kernel. The following chart shows the utilization of each multiprocessor during execution of the kernel.



3. Compute Resources

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

3.1. Kernel Profile - Instruction Execution

The Kernel Profile - Instruction Execution shows the execution count, inactive threads, and predicated threads for each source and assembly line of the kernel. Using this information you can pinpoint portions of your kernel that are making inefficient use of compute resource due to divergence and predication.

Examine portions of the kernel that have high execution counts and inactive or predicated threads to identify optimization opportunities.

Cuda Fuctions :
run_lbm(double[27]*, double[27]*, unsigned long, double, IndexBlock)

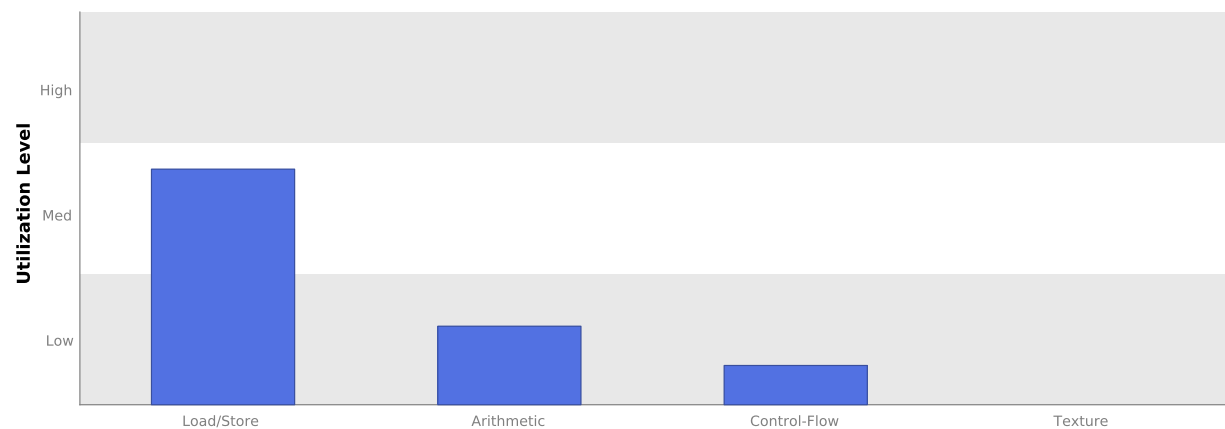
Maximum instruction execution count in assembly: 7830
Average instruction execution count in assembly: 3007
Instructions executed for the kernel: 2947560
Thread instructions executed for the kernel: 91476000
Non-predicated thread instructions executed for the kernel: 84960900
Warp non-predicated execution efficiency of the kernel: 90.1%
Warp execution efficiency of the kernel: 97.0%

Source files :
/home/egoh6/final/parallel/lbm_cuda.cu

3.2. 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.

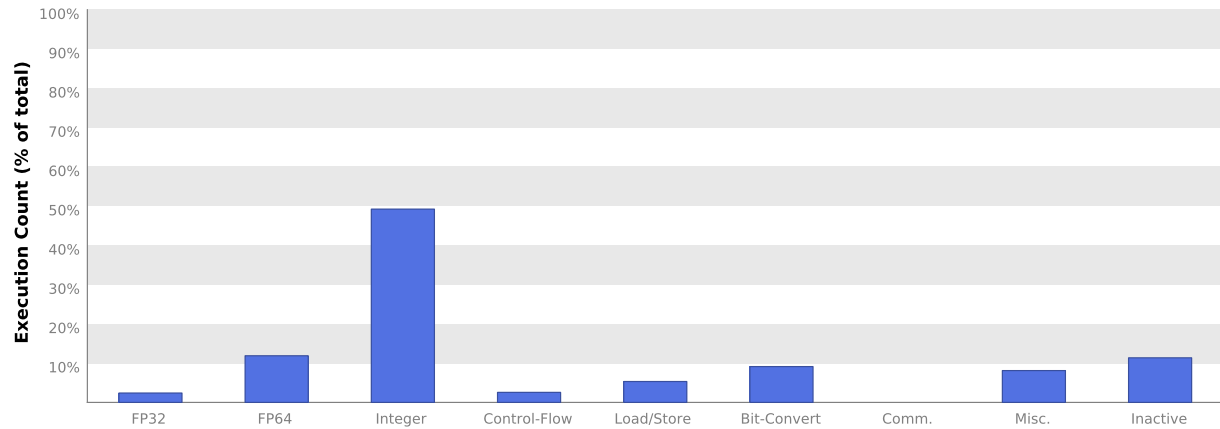
- Load/Store - Load and store instructions for local, shared, global, constant, etc. memory.
- Arithmetic - All arithmetic instructions including integer and floating-point add and multiply, logical and binary operations, etc.
- Control-Flow - Direct and indirect branches, jumps, and calls.
- Texture - Texture operations.



3.3. Instruction Execution Counts

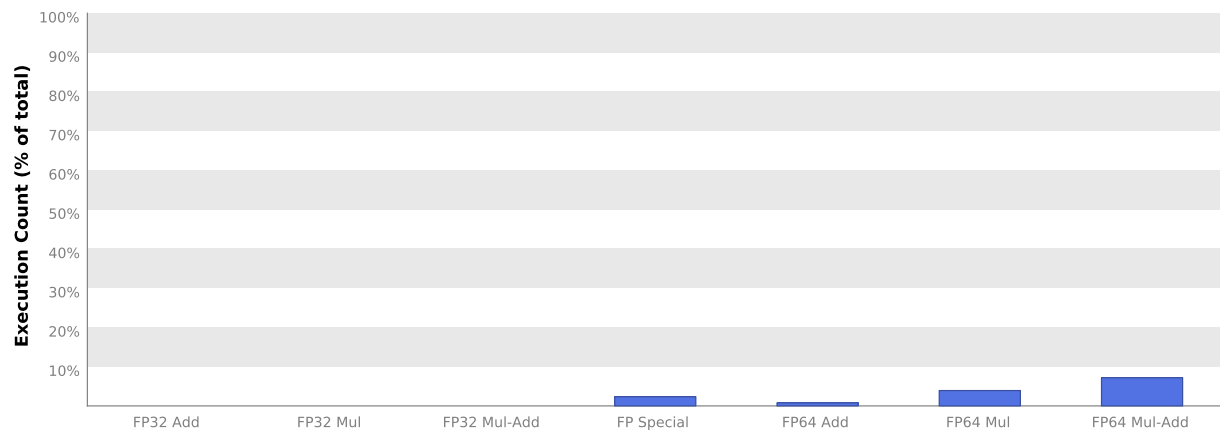
The following chart shows the mix of instructions executed by the kernel. The instructions are grouped into classes and for each class the chart shows the percentage of thread execution cycles that were devoted to executing instructions in that class. The

"Inactive" result shows the thread executions that did not execute any instruction because the thread was predicated or inactive due to divergence.



3.4. 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.



4. 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.

4.1. Memory Bandwidth And Utilization

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.

| Transactions | Bandwidth | Utilization | |
|--|-----------|-------------|--|
| L1/Shared Memory | | | |
| Local Loads | 0 | 0 B/s | |
| Local Stores | 0 | 0 B/s | |
| Shared Loads | 0 | 0 B/s | |
| Shared Stores | 0 | 0 B/s | |
| Global Loads | 1458000 | 7.936 GB/s | |
| Global Stores | 1458000 | 7.936 GB/s | |
| Atomic | 0 | 0 B/s | |
| L1/Shared Total | 2916000 | 15.873 GB/s | |
| L2 Cache | | | |
| L1 Reads | 1458000 | 7.936 GB/s | |
| L1 Writes | 1458000 | 7.936 GB/s | |
| Texture Reads | 0 | 0 B/s | |
| Atomic | 0 | 0 B/s | |
| Total | 2916000 | 15.873 GB/s | |
| Texture Cache | | | |
| Reads | 0 | 0 B/s | |
| Device Memory | | | |
| Reads | 1336908 | 7.277 GB/s | |
| Writes | 1382322 | 7.525 GB/s | |
| Total | 2719230 | 14.802 GB/s | |
| System Memory | | | |
| [PCIe configuration: Gen2 x16, 5 Gbit/s] | | | |
| Reads | 0 | 0 B/s | |
| Writes | 5 | 27.216 kB/s | |