Page: Summary | Result: 2 - 131 - d_bm | Add Baseline | Apply Rules | Occupancy Calculator | Source Comparison | Save as PDF

	Result	Time	Cycles	Regs	GPU	SM Frequency	CC	Process
Current	131 - d_bm (391, 1, 1)x(256, 1, 1)	13.37 msecond	2,08,10,661	168	0 - NVIDIA GeForce RTX 3090 Ti	1.56 cycle/nsecond	8.6	[6025] Decrypt_Serial

This table shows all results in the report. Use the column headers to sort the results in this report. Double-click a result to see detailed metrics.

ID	Estimated Speedup	Function Name	Demangled Name	Duration	Runtime Improvement (4.97879e+08)	Compute Throughput	Memory Throughput	# Registers	Grid Size	Block Size
0	88.58	initRAndComputeC	initRAndComputeC...	6.03	5.34	12.29	80.06	16	6250, 218, ...	16, 16,
1	58.54	matrixVectorMulKe...	matrixVectorMulKe...	171.46	100.37	93.15	58.19	37	6250, 8, ...	16, 16,
2	84.39	d_bm	d_bm(unsigned sho...	13.37	11.28	43.44	44.46	168	391, 1, ...	256, 1,
3	83.33	d_root	d_root(unsigned sh...	457.06	380.88	11.58	2.44	33	14, 1, ...	256, 1,

The following performance optimization opportunities were discovered for this result. Follow the rule links to see more context on the Details page.
Note: *Speedup estimates provide upper bounds for the optimization potential of a kernel assuming its overall algorithmic structure is kept unchanged.*

Uncoalesced Global Accesses
Est. Speedup: 84.39%
This kernel has uncoalesced global accesses resulting in a total of 591093750 excessive sectors (94% of the total 630500000 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.

Theoretical Occupancy
Est. Speedup: 55.54%
The 2.00 theoretical warps per scheduler this kernel can issue according to its occupancy are below the hardware maximum of 12. This kernel's theoretical occupancy (16.7%) is limited by the number of required registers.

Long Scoreboard Stalls
Est. Speedup: 49.08%
On average, each warp of this kernel spends 2.4 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 49.1% of the total average of 4.8 cycles between issuing two instructions.