**Device Name: Tesla C2075   
Device Revision Number: 2.0   
Global Memory Size: 5 636 554 752   
Number of Multiprocessors: 14   
Number of Cores: 448**   
Concurrent Copy and Execution: Yes   
**Total Constant Memory: 65 536**   
**Total Shared Memory per Block: 49 152**   
**Registers per Block: 32 768   
Warp Size: 32   
Maximum Threads per Block: 1024   
Maximum Block Dimensions: 1024, 1024, 64   
Maximum Grid Dimensions: 65535 x 65535 x 65535**Maximum Memory Pitch: 2147483647B   
Texture Alignment: 512B   
Clock Rate: 1147 MHz   
Execution Timeout: No   
Integrated Device: No   
Can Map Host Memory: Yes   
Compute Mode: default   
Concurrent Kernels: Yes   
ECC Enabled: Yes   
Memory Clock Rate: 1566 MHz   
Memory Bus Width: 384 bits   
L2 Cache Size: 786432 bytes   
Max Threads Per SMP: 1536   
Async Engines: 2   
Unified Addressing: Yes   
Initialization time: 310428 microseconds   
Current free memory: 5570056192   
Upload time (4MB): 2278 microseconds ( 713 ms pinned)   
Download time: 1428 microseconds ( 697 ms pinned)   
Upload bandwidth: 1841 MB/sec (5882 MB/sec pinned)   
Download bandwidth: 2937 MB/sec (6017 MB/sec pinned)

- Compute capatibility: 2.0

- Warp size: 32

- Maximum number of resident blocks per multiprocessor: 8

- Maximum number of resident warps per multiprocessor: 48

- Maximum number of resident threads per multiprocessor: 1536

- Number of 32-bit registers per multiprocessor: 32 KB

- Maximum number of 32-bit registers per thread: 63

- Maximum amount of shared memory per multiprocessor: 48 KB

- Constant memory size: 64 KB

- One of the keys to good performance is to keep the multiprocessors on the device as busy as possible.

- Hardware utilization can also be improved in some cases by designing your application so that multiple, independent kernels can execute at the same time.

- Occupancy = the number of active warps/multiprocessor to the maximum number of possible active warps.

- Choosing grid-size: The number of blocks in a grid should be larger than the number of multiprocessors so that all multiprocessors have at least one block to execute.

# blocks = A x # multiprocessors = 14 x const

- Choosing block-size: multiple concurrent blocks can reside on a multiprocessor.

* Threads per block should be a multiple of warp size (32) to avoid wasting computation on under-populated warps and to facilitate coalescing.

**# thread = 32 x const**

* A minimum of 64 threads per block should be used, and only if there are multiple concurrent blocks per multiprocessor.

**# thread ≥ 64**

* Between 128 and 256 threads per block is a better choice and a good initial range for experimentation with different block sizes.

**128 ≤ # thread ≤ 256**

* Use several (3 to 4) smaller thread blocks rather than one large thread block per multiprocessor if latency affects performance. This is particularly beneficial to kernels that frequently call \_\_syncthreads().

Note that when a thread block allocates more registers than are available on a multiprocessor, the kernel launch fails, as it will when too much shared memory or too many threads are requested.