Fermi architecture.

- The G80 Architecture.

① the first GPU to support C

② the first GPU to replace the separate vertex and pixel pipelines with a single unified processor that executed vertex, geometry, pixel, and computing programs.

③ the first GPU to utilize a scalar thread processor eliminating the need for programers to manually manage vector register.

scalar thread processor
↕
Vector processor

④ Introduced the single instruction multiple thread (SIMT) execution model.

⑤ introduced shared memory and barrier synchronization for inter-thread communication.

- Fermi

① Improve double precision performance.
② ECC support (memory error 해결)
③ True cache hierarchy (parallel algorithms 중에 GPU의 shared memory를 사용 못하는 경우를 위하여 각각)
④ More shared memory (SM shared memory 를 (64KB 사용가능한 application 만듦).
⑤ Faster context switching, Atomic operations.

- Third Generation Streaming Multiprocessor
- 32 CUDA cores per SM
- Dual Warp scheduler
  두개 warps를 동시에 효율적으로 schedule 하고 dispatch 함
- 64KB of RAM을 shared memory 와 L1 cache로 설정함
  (∵ shared memory + L1 cache = 64KB)

- Second Generation Parallel Thread Execution ISA
  - Memory access instructions to support transition to 64 bit addressing
  - Improved performance through Prediction.

- Improved Memory Subsystem
  - NVIDIA Parallel Data Cache hierarchy with Configurable L1 and Unified L2 Caches
  - Greatly improved atomic memory operation performance
    (read → modify → write)

- Nvidia GigaThread engine
  - faster context switching
  - concurrent kernel execution
  - out of order thread block execution

A Quick Refresher on CUDA

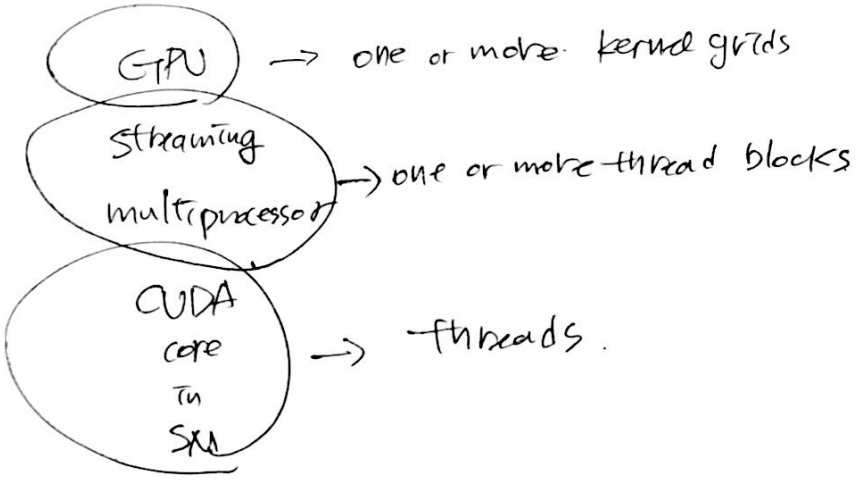
- CUDA
  the hardware and software architecture that enables NVIDIA GPUs to execute programs written with C, C++, Fortran, OpenCL, DirectCompute and other languages.
- CUDA program → parallel kernels.
- kernel → parallel threads들이 각각 parallel하게 수행함.
- 프로그래머가 각각의 이런 threads들을 threadblock과 grid of thread blocks로 나눔.
- GPU instantiates a kernel program on a grid of parallel thread blocks.
- thread block & kernel이 어떻게 run
- thread
  ① thread ID
  ② program counter
  ③ registers.
  ④ per-thread private memory.
  ⑤ input + output.
- thread block : 같이 협동하는 threads 들의 집합. block ID 가짐.
  using
  ① barrier synchronization : 한 지점에서 모든 thread 다 도착할 때까지 더 이상 진행 X.
  ② shared memory.
  has ~~thread ID~~ or within its grid!
- Grid
  - an array of thread blocks that execute the same kernel
  - input from, output to global memory.
  - synchronize between dependent kernel calls.
- thread
  per-thread private memory space.
  ① register spills
  ② function calls
  ③ C automatic array variables (C dynamic allocation?)
- thread block
  per-block shared memory space
  ① fine-grain thread block communication
  ② data sharing
  ③ result sharing
- thread blocks grid
  Global Memory space
  ① share results

Hardware Execution.

CUDA's hierarchy of threads

↕

hierarchy of processors on the GPU.



SM : 32 threads == Warp

An Overview of The Fermi Architecture

- 512 CUDA cores.

$= 16_{2^4}$ SM × 32_{2^5} CUDA cores per SM

- 64 bit memory partitions
- 384 bit memory interface.
  ⇒ 6 GB of GDDR5 DRAM Max.

- Giga Thread global scheduler
  distributes thread blocks
  to SM thread Schedulers.

512 high performance CUDA cores

- 1 SM 32 CUDA cores
- CUDA core ALU + FPU pipelined
- IEEE 754-2008 floating point standard
- fused multiply-add (FMA) instruction.
- FMA improves over a multiply-add (MAD) instruction for both single and double precision arithmetic
  → precision 좋아
- 32 bit ALU에서 32bit instruction을 만족함
- 64 bit extended precision operation. 지원

16 Load/Store Units.

- Source and destination addresses to be calculated for sixteen threads per clock, each
- cache or DRAM

4 Special Function Units. (SFUs)

- execute transcendental instructions ex) sin, cosine, reciprocal, square root ....
- one instruction per thread per clock
  warp는 8 clocks 이상!
- The SFU pipeline is decoupled from the dispatch unit, allowing the dispatch unit to issue to other execution units while the SFU is occupied

Dual Warp Scheduler

- 1 SM (32 parallel threads) = (1 warps)
- 1 SM 2 Warp scheduler
  2 instruction dispatch units.

→ 두개의 warps 가 동시에 다뤄짐

- dual warp scheduler
  ① Warps 2개 고름
  ② 각 warp에서 한 instruction을
  16 cores, 16 load/store units, or 4 SFU 에 dispatch 함.
  ③ warps 간에는 독립적 → scheduler 가 instruction stream 안의 dependency check 필요X.
  ④ 같이 dispatch 된 instruction은 dual issued
  2 integer 2 floating point. mixed integer, floating point, load/store, SFU ... 등등에 다양하게 가능.
  ⑤ Double precision instructions do not support dual dispatch with any other operation.

64kB Configurable Shared Memory and L1 Cache

- On chip Shared memory.
- 같은 thread block 간에 reuse of on-chip data ↑
  off-chip traffic ↓
- 64kB on chip memory
  = 48kB shared memory
  +
  16kB L1 cache
  or
  16kB Shared memory.
  +
  48kB L1 cache.