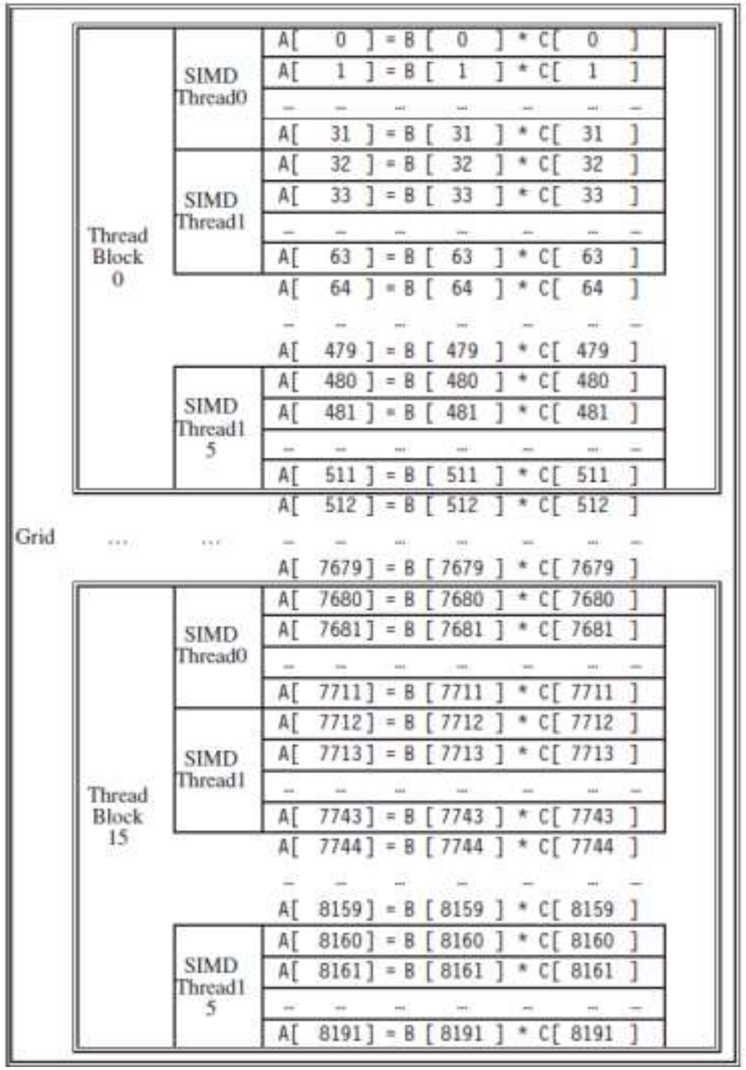
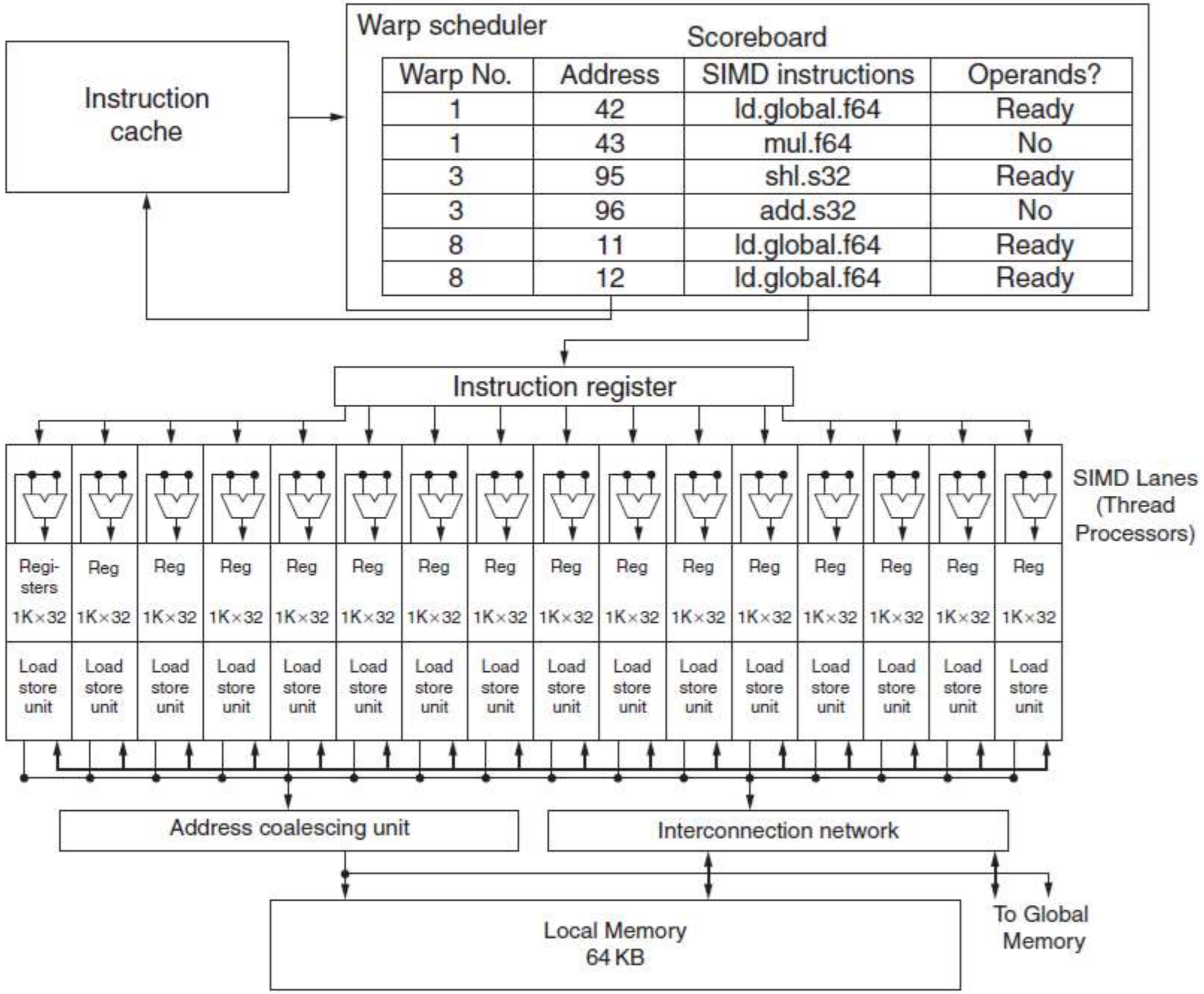
Vector-Vector Multiply Example

- For a vector-vector multiply with 8192 elements:
  - The Grid (vectorizable loop) works on all 8192 elements
  - With 512 elements per Thread Block, we need 16 Thread Blocks
  - Each Thread Block contains 16 threads of SIMD instructions (SIMD Threads)
  - Each thread of SIMD instructions calculates 32 elements per instruction
- The Thread Block Scheduler assigns Thread Blocks to multithreaded SIMD Processors, and the Thread Scheduler picks which thread of SIMD instructions to run each clock cycle.



Multithreaded SIMD Processor



- 16 SIMD lanes for parallel execution
- SIMD Thread (Warp) Scheduler with ~48 independent threads
- Scoreboard to track which threads are ready to run
- Dispatch unit to send threads to the processor

GPU Hardware Schedulers

1

Thread Block Scheduler

Assigns Thread Blocks (bodies of vectorized loops) to multithreaded SIMD Processors

Ensures thread blocks are assigned to processors whose local memories have the corresponding data

2

SIMD Thread Scheduler

Operates within a SIMD Processor

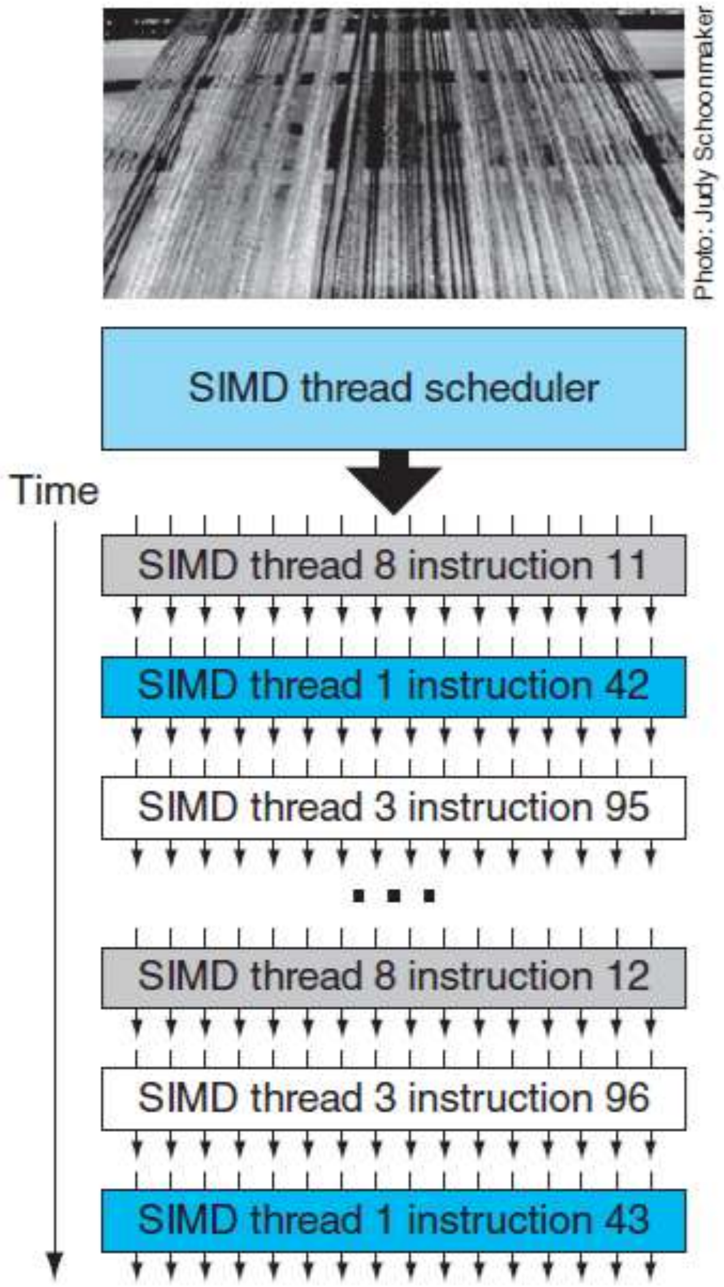
Schedules when threads of SIMD instructions should run

Includes a scoreboard to track up to 48 threads of SIMD instructions

These two levels of hardware schedulers work together to manage the execution of thousands of threads across multiple SIMD processors.

SIMD Thread Scheduling

- The SIMD Thread Scheduler selects a ready thread of SIMD instructions and issues an instruction synchronously to all the SIMD Lanes executing that thread.
- Because threads of SIMD instructions are independent, the scheduler may select a different SIMD thread each time, allowing it to hide memory latency and increase processor utilization.
- This approach differs from vector processors, which typically execute a vector instruction to completion before starting the next one.



GPU Register Organization

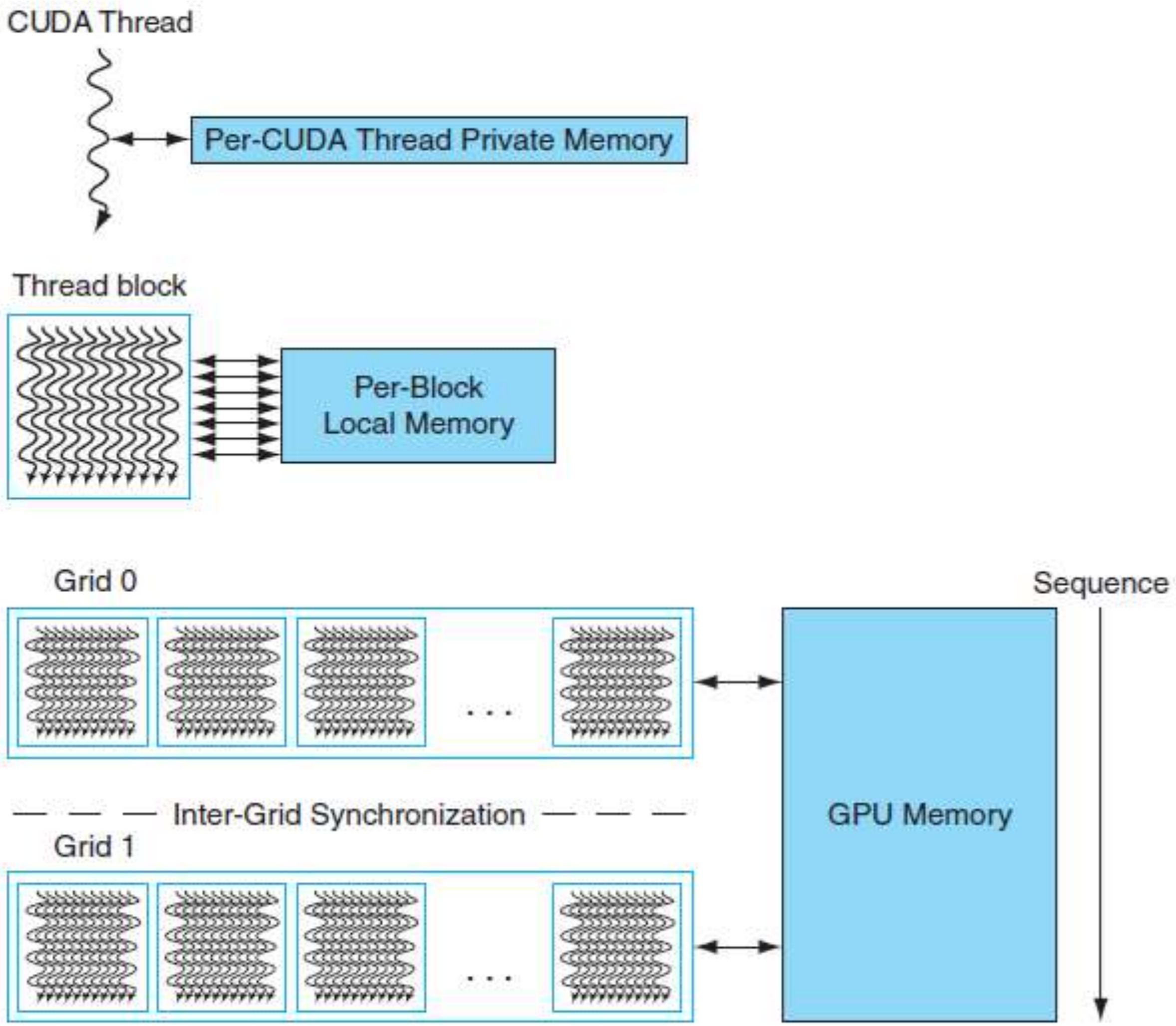
Register Capacity

- 32,768 32-bit registers per SIMD Processor
- Registers divided logically across SIMD Lanes
- Each SIMD Thread limited to no more than 64 registers
- With 16 physical SIMD Lanes, each contains 2048 registers

Register Usage

- Think of a SIMD Thread as having up to 64 vector registers
- Each vector register has 32 elements (32 bits wide each)
- Double-precision operands use two adjacent 32-bit registers
- Registers dynamically allocated when threads are created
- Registers freed when the SIMD Thread exits

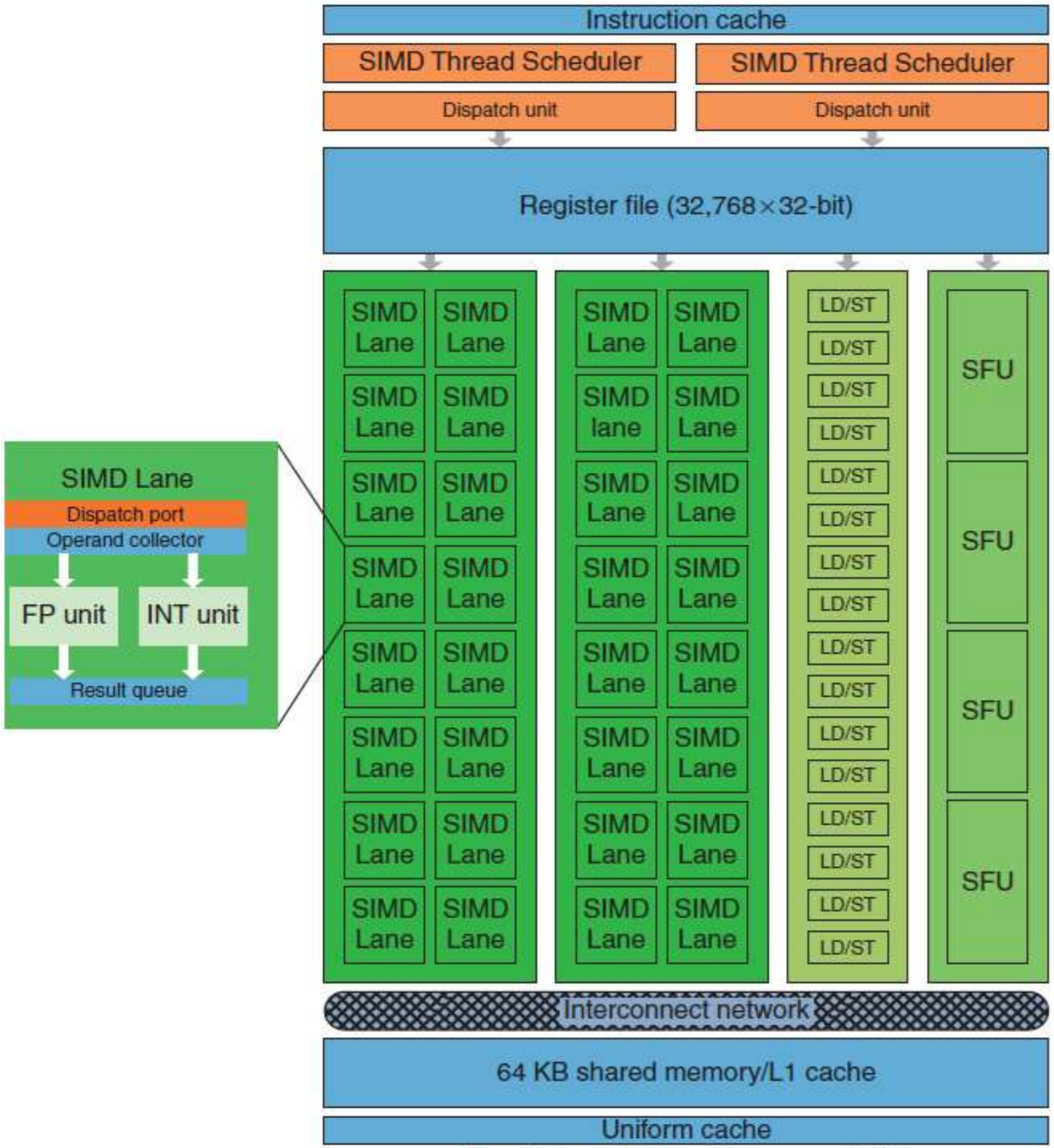
GPU Memory Structures



- Private Memory
  - Off-chip DRAM private to each SIMD Lane
  - Used for stack frames, register spilling, and private variables
  - Not shared between SIMD Lanes
- Local Memory (Shared Memory)
  - On-chip memory local to each multithreaded SIMD Processor
  - Shared by SIMD Lanes within a processor, but not between processors
  - Dynamically allocated to thread blocks
- GPU Memory (Global Memory)
  - Off-chip DRAM shared by the whole GPU and all thread blocks
  - Accessible by the host (system processor)

Fermi Architecture

- L1 Data and Instruction Cache for each SIMD Processor
- 768 KB L2 cache shared by all SIMD Processors
- Configurable SRAM: 16KB L1/48KB Local Memory or 48KB L1/16KB Local Memory



Fermi streaming multiprocessor (SM)