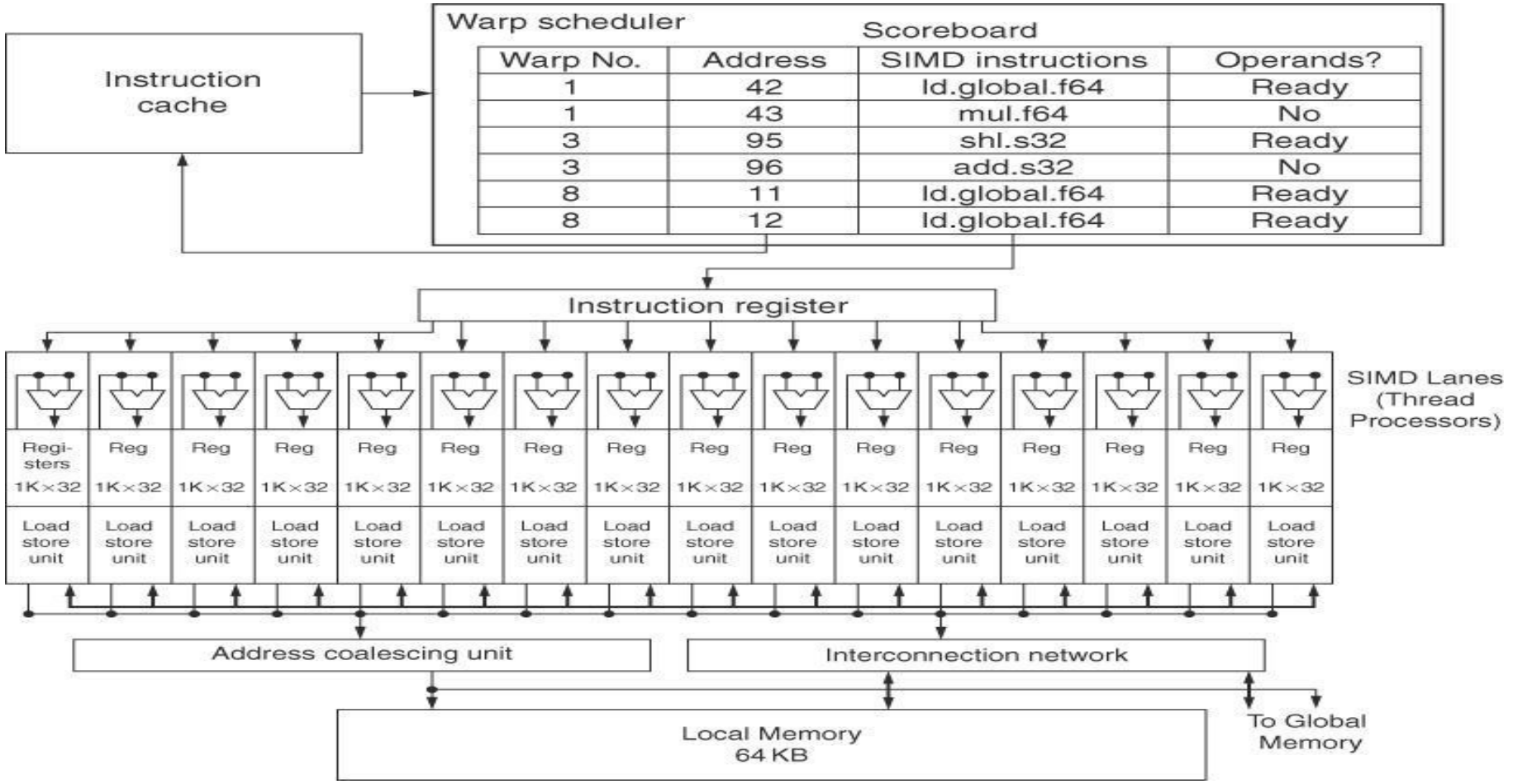
FERMI GPU ARCHITECTURE

NVIDIA GPU – MTSIMD



NVIDIA GPU- MTSIMD

- GPU is a multiprocessor composed of MTSIMD processors.
- It is similar to vector processor but with many parallel FU's that are deeply pipelined.
- MTSIMD is a processor that executes code in the form of thread blocks.
- GPU H/W contains a collection of MTSIMD Processors execute a Grid of Thread Blocks.

NVIDIA GPU- MTSIMD

- GPU H/W has two levels of H/W schedulers

1. Thread Block Scheduler:

  - Thread block scheduler is similar to control unit in Vector processor
  - determine the no of thread blocks for a loop and allocates them to different MTSIMD processors.
  - ensures that thread blocks are assigned to the processors whose local memories have the corresponding data.

NVIDIA GPU- FERMI MTSIMD

2. SIMD Thread Scheduler:

- SIMD Thread scheduler has scoreboard logic
- It keeps track of 48 threads of SIMD instructions
- It tells that which thread of SIMD instructions are ready to run
- It sends those instructions to dispatch unit to be run on MTSIMD processor
- within a SIMD Processor, which schedules when threads of SIMD instructions should run

NVIDIA GPU- MTSIMD

- It has many parallel functional units
- SIMD Processors with separate PCs and are programmed using threads.
- Each MTSIMD Processor is assigned 512 elements of the vectors to work on
- SIMD processors have 32,768 registers
- Like vector processor these registers are logically divided across SIMD lanes.

NVIDIA GPU- MTSIMD

- Each SIMD Thread has 64 vector registers of 32 elements with 32 bit each.
- FERMI has 16 physical lanes each contain 2048 registers
- Thread Blocks would contain 512/32 = 16 SIMD threads.
- Each thread of SIMD instructions in this example compute 32 of the elements of the computation.

NVIDIA GPU- MTSIMD

- GPU applications have so many threads of SIMD instructions that multithreading can
  - hide the latency to DRAM
  - increase utilization of multithreaded SIMD Processors

NVIDA GPU ISA

- PTX(Parallel Thread Execution) provides a stable instruction set for GPUs
- H/W instruction set is hidden from the programmer
- PTX instructions describe the operations on a single CUDA thread
- PTX uses virtual registers
- Translation to machine code is performed in software

NVIDA GPU ISA

- Format of a PTX instruction is

opcode.type d, a, b, c;

– where d is the destination operand; a, b, and c are source operands

- Source operands are 32-bit or 64-bit registers or a constant value.
- Destinations are registers,except for store instructions.

SSN

NVIDA GPU ISA

- the operation type is one of the following:

Type	.type Specifier			
• Untyped bits 8, 16, 32, and 64 bits	. b8,	b16,	. b32,	b64
• Unsigned integer 8, 16, 32, and 64 bits	.U8,	. U16,	U32,	u64
• Signed integer 8, 16, 32, and 64 bits	. S8,	. S16,	. S32,	S64
• Floating Point 16, 32, and 64 bits	.J16, J32, J64			

Conditional Branching

- Like vector architectures, GPU branch hardware uses internal masks
- Also uses
  - Branch synchronization stack
  - Entries consist of masks for each SIMD lane
  - i.e. which threads commit their results

- Per-thread-lane 1-bit predicate register, specified by programmer