NVIDIA GPU Computational Structures

- We use NVIDIA systems as our example as they are representative of GPU architectures. Specifically, we follow the terminology of the CUDA parallel programming language above and use the Fermi architecture.

- Like vector architectures, GPUs work well only with data-level parallel problems. Both styles have gather-scatter data transfers and mask registers, and GPU processors have even more registers than do vector processors. Since they do not have a close-by scalar processor, GPUs sometimes implement a feature at runtime in hardware that vector computers implement at compiler time in software. Unlike most vector architectures, GPUs also rely on multithreading.

- A Grid is the code that runs on a GPU that consists of a set of Thread Blocks. Figure below draws the analogy between a grid and a vectorized loop and between a Thread Block and the body of that loop.

Type	More descriptive name	Closest old term outside of GPUs	Official CUDA/ NVIDIA GPU term	Book definition
Program abstractions	Vectorizable Loop	Vectorizable Loop	Grid	A vectorizable loop, executed on the GPU, made up of one or more Thread Blocks (bodies of vectorized loop) that can execute in parallel.
	Body of Vectorized Loop	Body of a (Strip-Mined) Vectorized Loop	Thread Block	A vectorized loop executed on a multithreaded SIMD Processor, made up of one or more threads of SIMD instructions. They can communicate via Local Memory.
	Sequence of SIMD Lane Operations	One iteration of a Scalar Loop	CUDA Thread	A vertical cut of a thread of SIMD instructions corresponding to one element executed by one SIMD Lane. Result is stored depending on mask and predicate register.
Machine object	A Thread of SIMD Instructions	Thread of Vector Instructions	Warp	A traditional thread, but it contains just SIMD instructions that are executed on a multithreaded SIMD Processor. Results stored depending on a per-element mask.
	SIMD Instruction	Vector Instruction	PTX Instruction	A single SIMD instruction executed across SIMD Lanes.
Processing hardware	Multithreaded SIMD Processor	(Multithreaded) Vector Processor	Streaming Multiprocessor	A multithreaded SIMD Processor executes threads of SIMD instructions, independent of other SIMD Processors.
	Thread Block Scheduler	Scalar Processor	Giga Thread Engine	Assigns multiple Thread Blocks (bodies of vectorized loop) to multithreaded SIMD Processors.
	SIMD Thread Scheduler	Thread scheduler in a Multithreaded CPU	Warp Scheduler	Hardware unit that schedules and issues threads of SIMD instructions when they are ready to execute; includes a scoreboard to track SIMD Thread execution.
	SIMD Lane	Vector Lane	Thread Processor	A SIMD Lane executes the operations in a thread of SIMD instructions on a single element. Results stored depending on mask.
Memory hardware	GPU Memory	Main Memory	Global Memory	DRAM memory accessible by all multithreaded SIMD Processors in a GPU.
	Private Memory	Stack or Thread Local Storage (OS)	Local Memory	Portion of DRAM memory private to each SIMD Lane.
	Local Memory	Local Memory	Shared Memory	Fast local SRAM for one multithreaded SIMD Processor, unavailable to other SIMD Processors.
	SIMD Lane Registers	Vector Lane Registers	Thread Processor Registers	Registers in a single SIMD Lane allocated across a full thread block (body of vectorized loop).

Fig: Quick guide to GPU terms

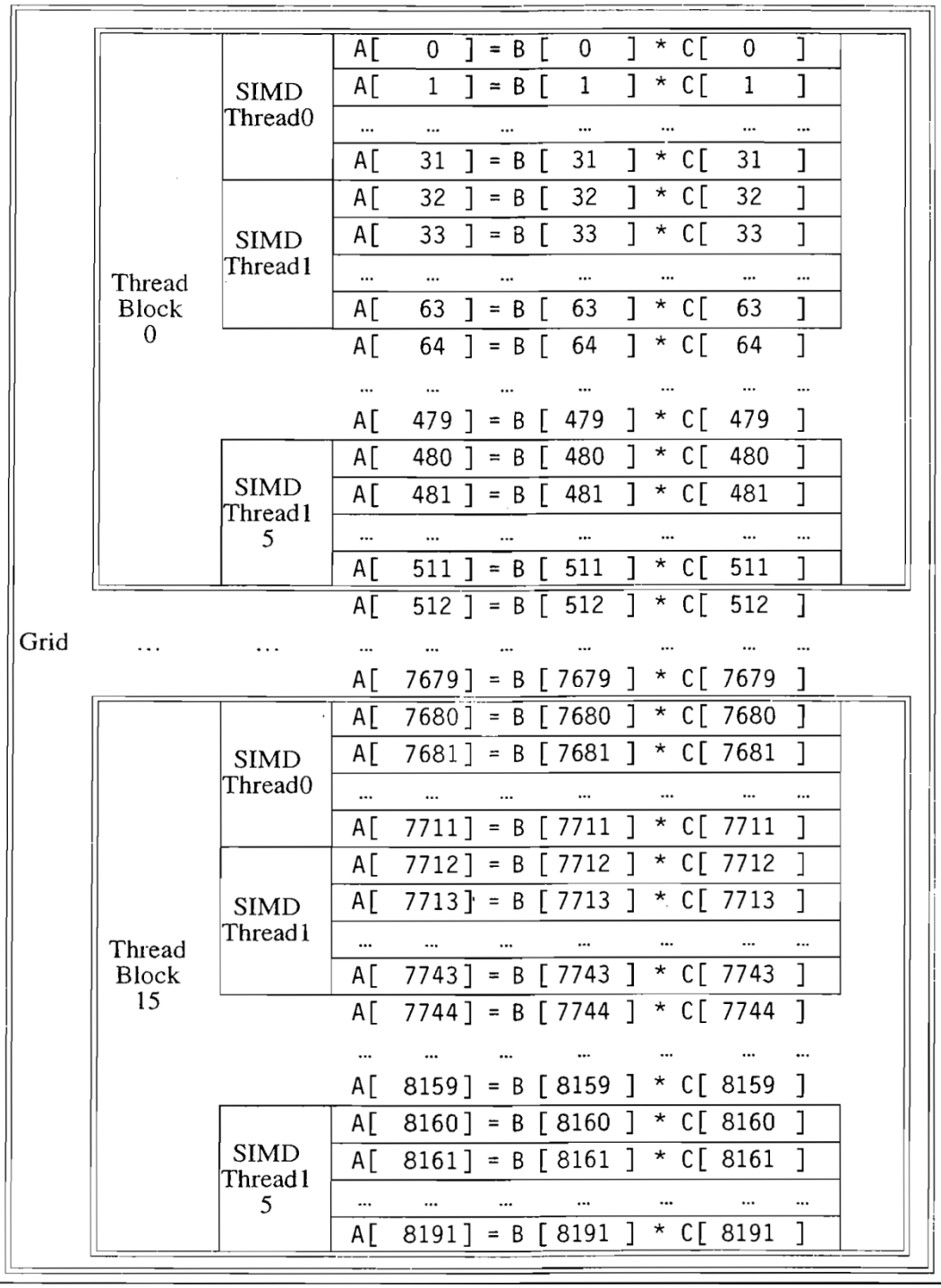

Fig: Mapping of Grid, Thread Block, and Threads of SIMD

- suppose we want to multiply two vectors together, each 8192 elements long. We'll return to this example throughout this section. Figure 4.13 shows the relationship between this example and these first two GPU terms. The GPU code that works on the whole 8192 element multiply is called a Grid (or vectorized loop). To break it down into more manageable sizes, a Grid is composed of Thread Blocks (or body of a vectorized loop), each with up to 512 elements. Note that a SIMD instruction executes 32 elements at a time. With 8192 elements in the vectors, this example thus has 16 Thread Blocks since 16 = 8192 _ 512.