William Stallings Computer Organization and Architecture 10th Edition

Chapter 19

General-Purpose Graphic Processing Units

Compute Unified Device Architecture (CUDA)

- A parallel computing platform and programming model created by NVIDIA and implemented by the graphics processing units (GPUs) that they produce
- CUDA C is a C/C++ based language
- Program can be divided into three general sections
  - Code to be run on the host (CPU)
  - Code to be run on the device (GPU)
  - The code related to the transfer of data between the host and the device
- The data-parallel code to be run on the GPU is called a *kernel*
  - Typically will have few to no branching statements
  - Branching statements in the kernel result in serial execution of the threads in the GPU hardware
- A *thread* is a single instance of the kernel function
  - The programmer defines the number of threads launched when the kernel function is called
  - The total number of threads defined is typically in the thousands to maximize the utilization of the GPU processor cores, as well as maximize the available speedup
  - The programmer specifies how these threads are to be bundled

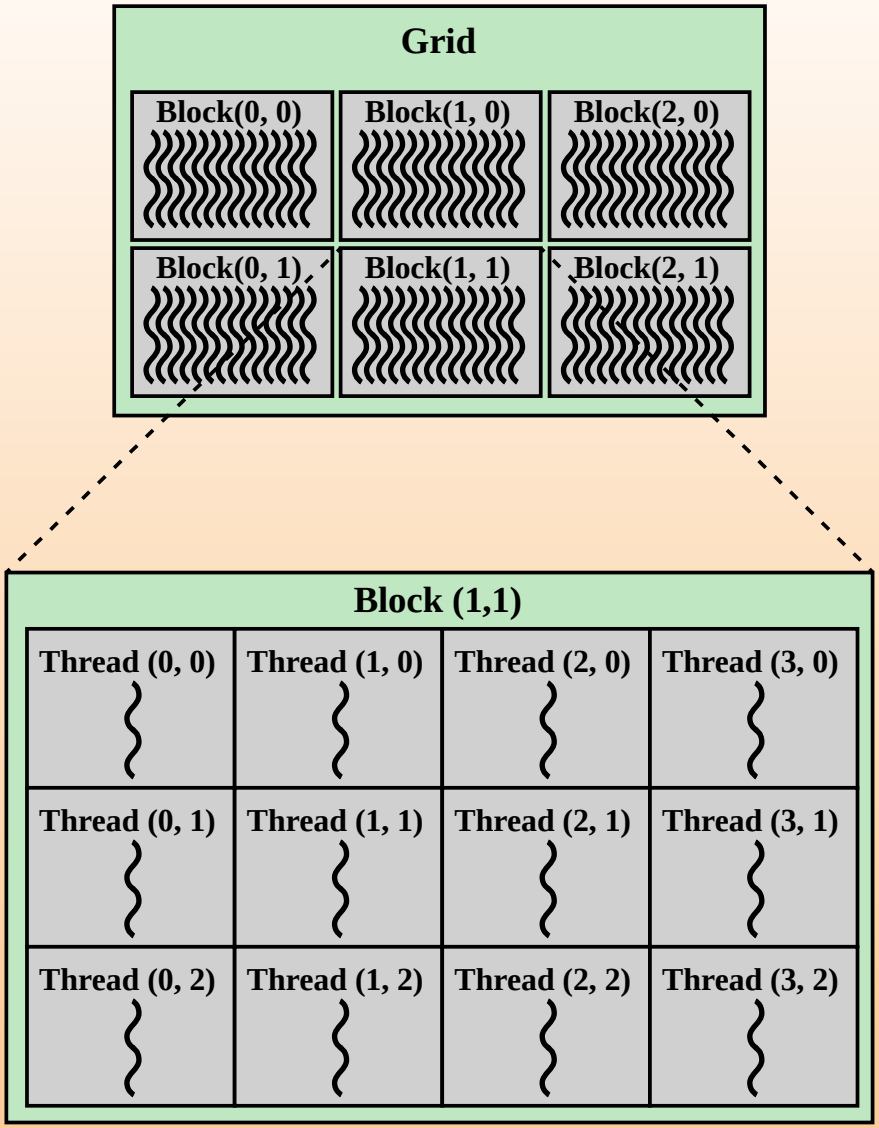


Figure 19.1 Relationship Among Threads, Blocks, and a Grid

Table 19.1

CUDA Terms to GPU's Hardware Components Equivalence Mapping

CUDA Term	Definition	Equivalent GPU Hardware Component
Kernel	Parallel code in the form of a function to be run on GPU	Not applicable
Thread	An instance of the kernel on the GPU	GPU/CUDA processor core
Block	A group of threads assigned to a particular SM	CUDA multiprocessor (SM)
Grid	The GPU	GPU

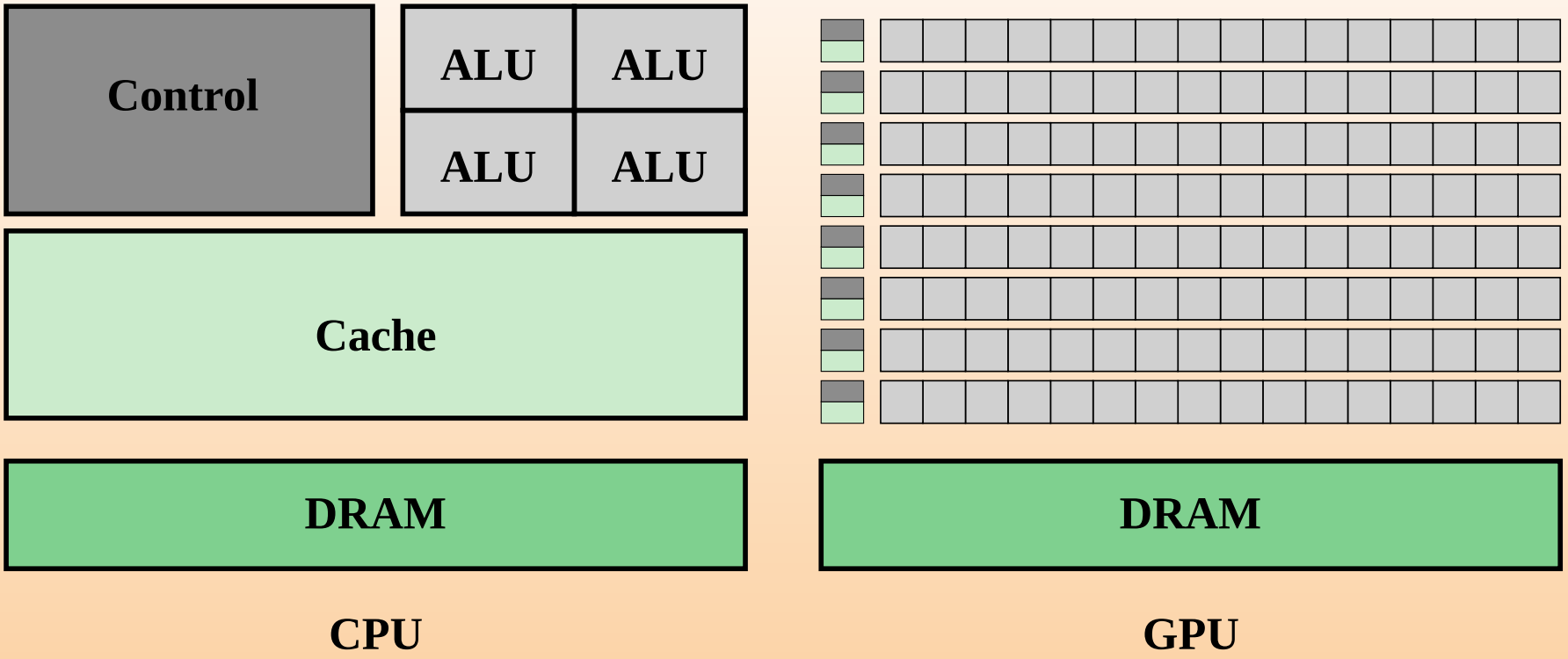


Figure 19.2 CPU vs. GPU Silicon Area/Transistor Dedication

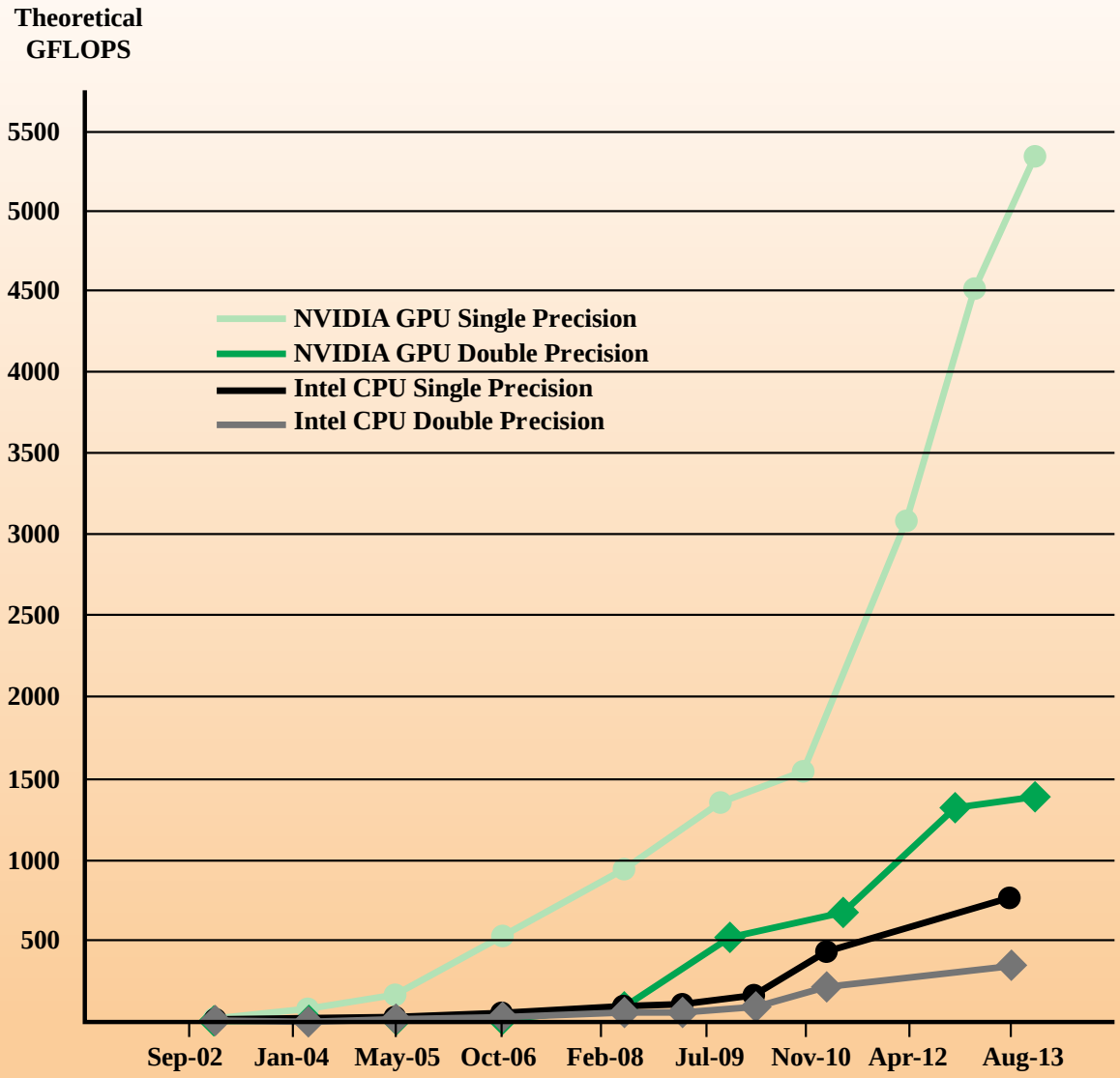


Figure 19.3 Floating-Point Operations per Second for CPU and GPU

GPU Architecture Overview

The historical evolution can be divided up into three major phases:

The first phase would cover early 1980s to late 1990s, where the GPU was composed of fixed, nonprogrammable, specialized processing stages

The second phase would cover the iterative modification of the resulting Phase I GPU architecture from a fixed, specialized, hardware pipeline to a fully programmable processor (early to mid-2000s)

The third phase covers how the GPU/GPGPU architecture makes an excellent and affordable highly parallelized SIMD coprocessor for accelerating the run times of some nongraphics-related programs, along with how a GPGPU language maps to this architecture

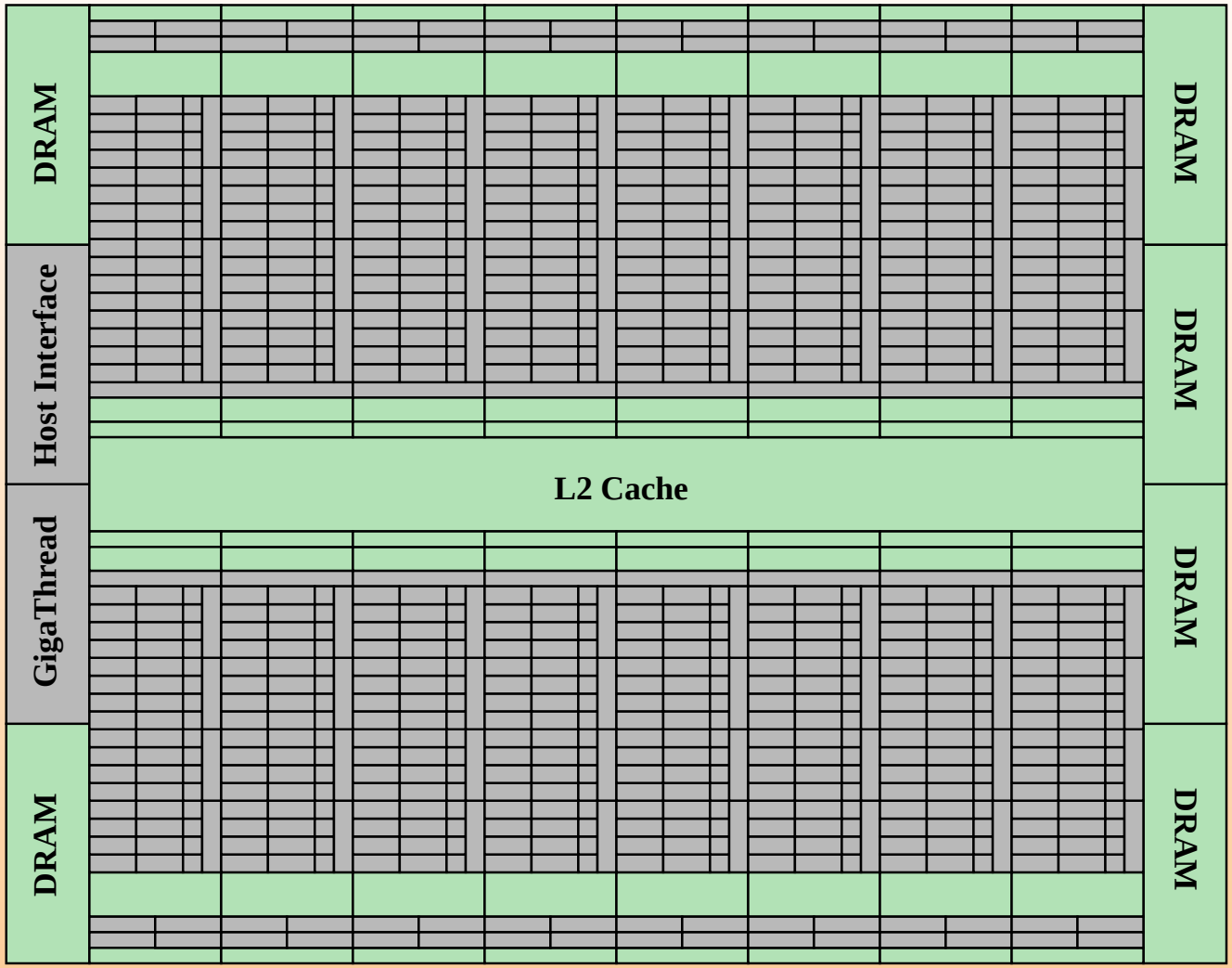


Figure 19.4 NVIDIA Fermi Architecture

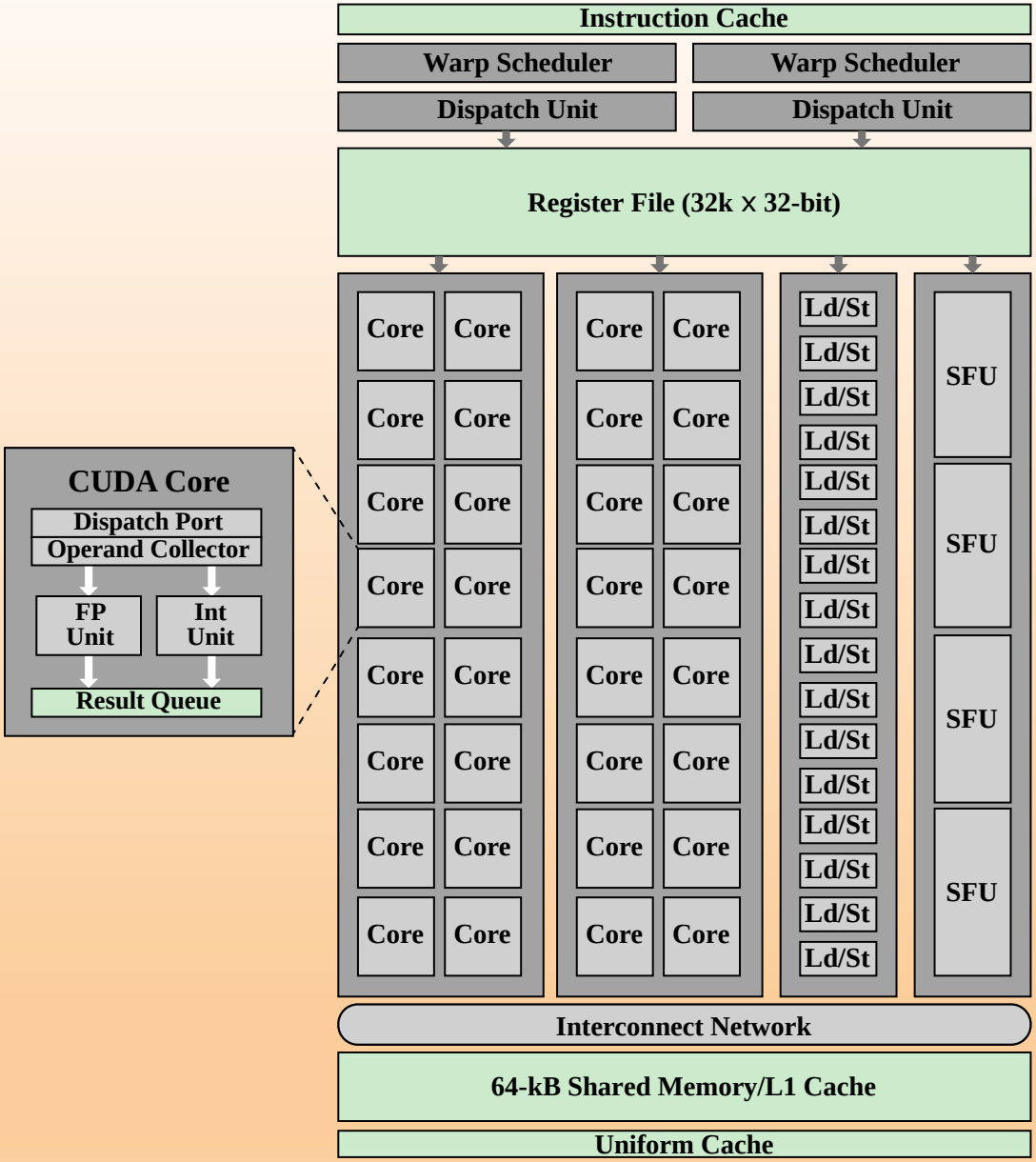


Figure 19.5 Single SM Architecture

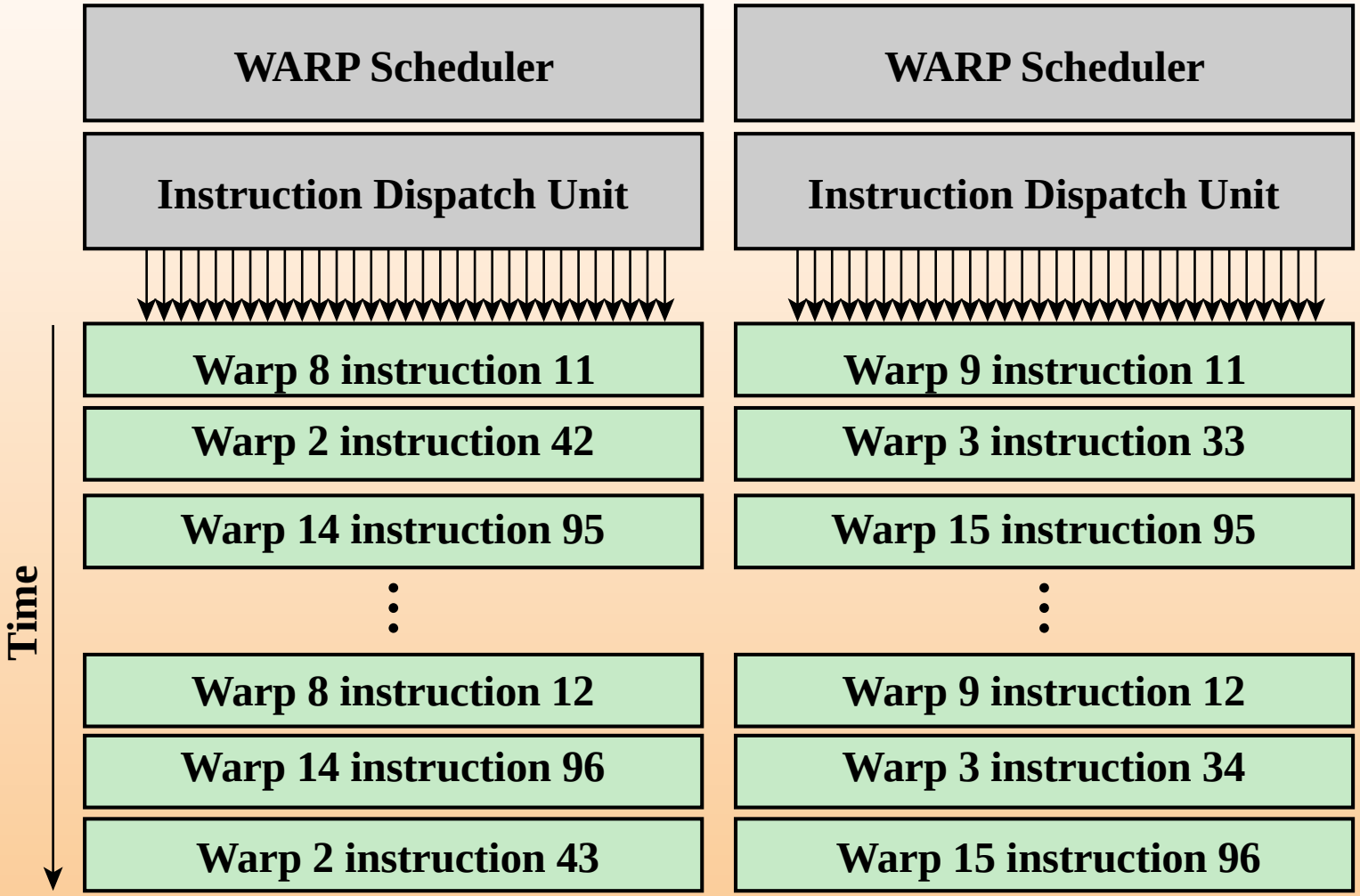


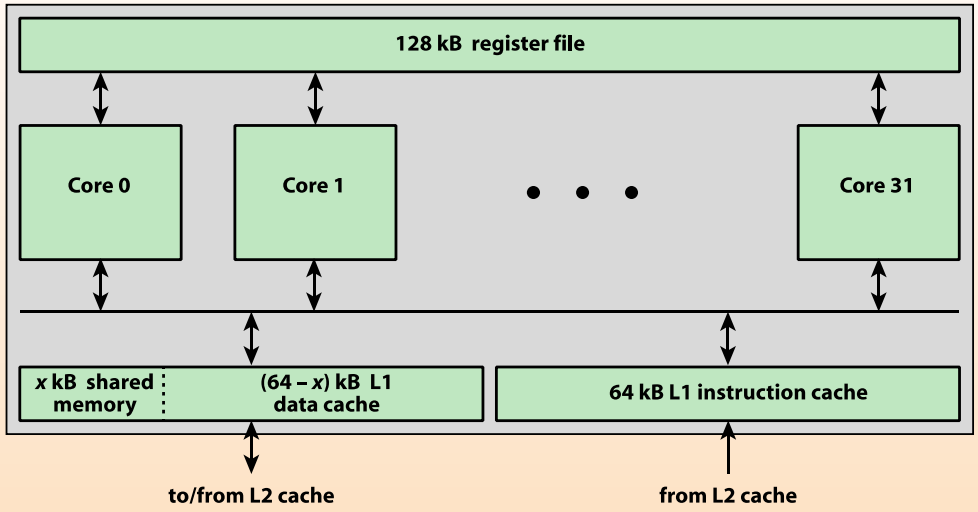
Figure 19.6 Dual Warp Schedulers and Instruction Dispatch Units Run Example

CUDA Cores

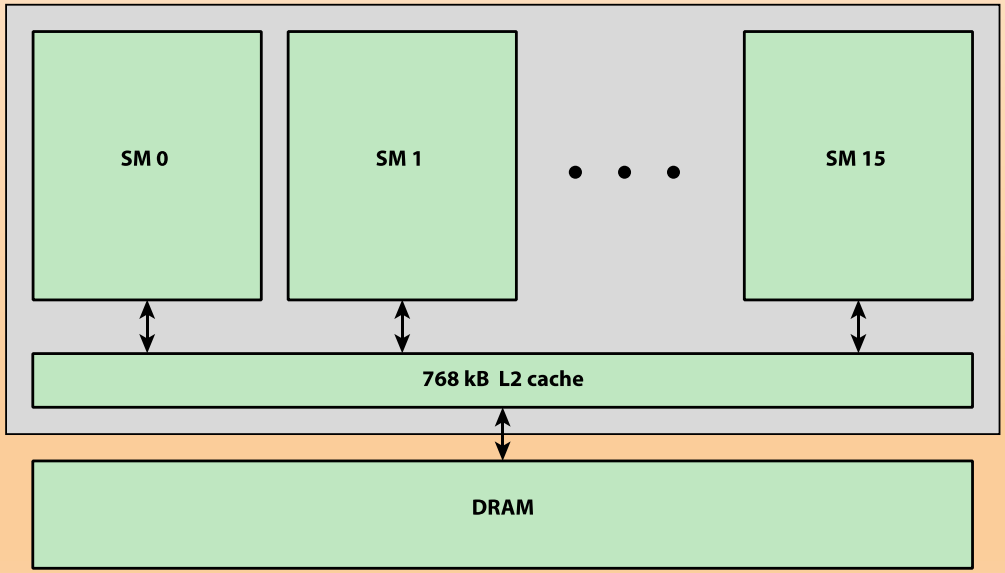
- The NVIDIA GPU processor cores are also known as CUDA cores
- There are a total of 32 CUDA cores dedicated to each SM in the Fermi architecture
- Each CUDA core has two separate pipelines or data paths
  - An integer (INT) unit pipeline
    - Is capable of 32-bit, 64-bit, and extended precision for integer and logic/bitwise operations
  - Floating-point (FP) unit pipeline
    - Can perform a single-precision FP operation, while a double-precision FP operation requires two CUDA cores

Table 19.2
GPU's Memory Hierarchy Attributes

Memory Type	Relative Access Times	Access Type	Scope	Data Lifetime
Registers	Fastest. On-chip	R/W	Single thread	Thread
Shared	Fast. On-chip	R/W	All threads in a block	Block
Local	100´ to 150´ slower than shared & register. Off-chip	R/W	Single thread	Thread
Global	100´ to 150´ slower than shared & register. Off-chip.	R/W	All threads & host	Application
Constant	100´ to 150´ slower than shared & register. Off-chip	R	All threads & host	Application
Texture	100´ to 150´ slower than shared & register. Off-chip	R	All threads & host	Application



(a) SM memory architecture



(b) Overal memory architecture

Figure 19.7 Fermi Memory Architecture

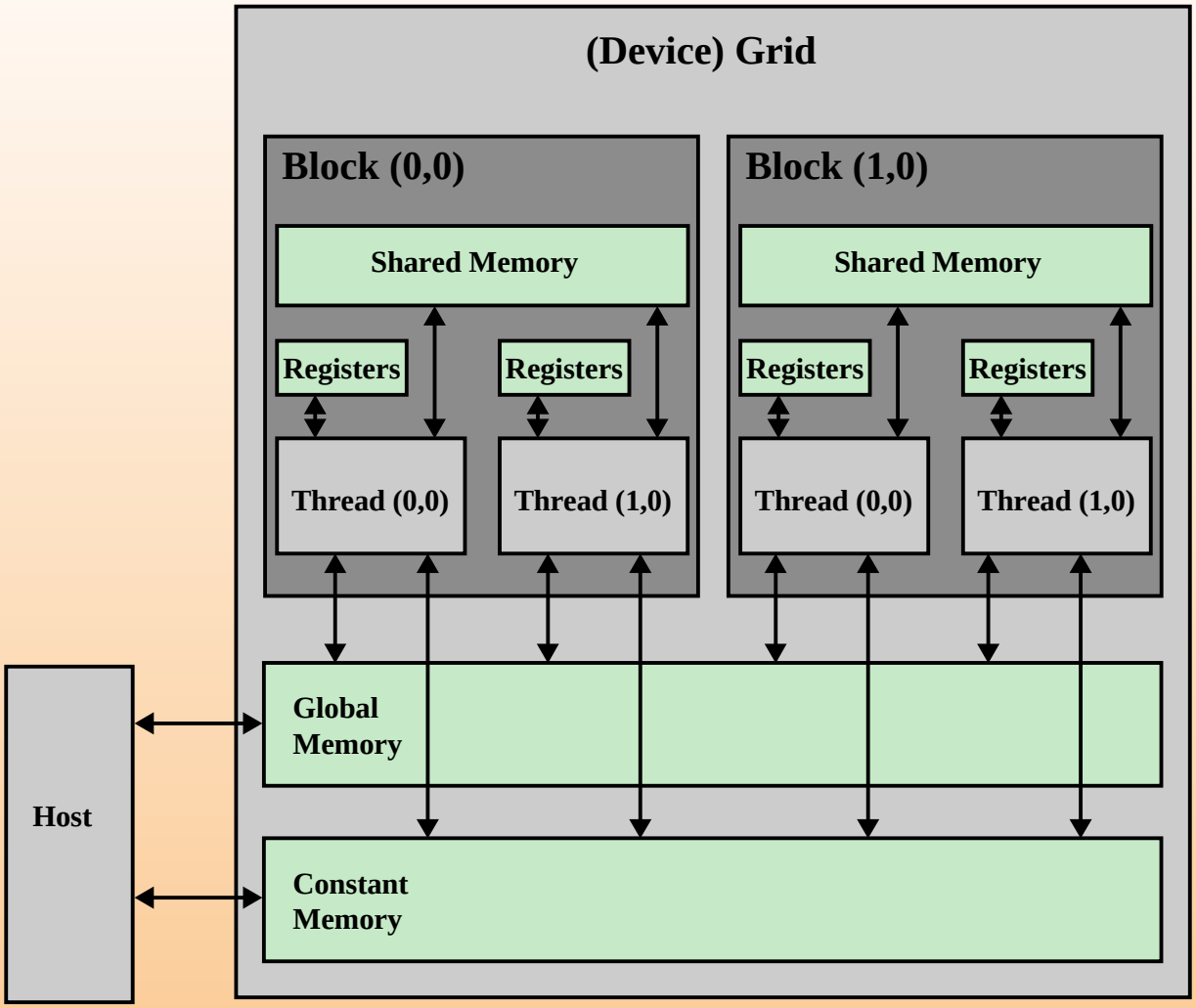


Figure 19.8 CUDA Representation of a GPU's Basic Architecture. The example GPU shown has two SMs and two CUDA cores per SM.

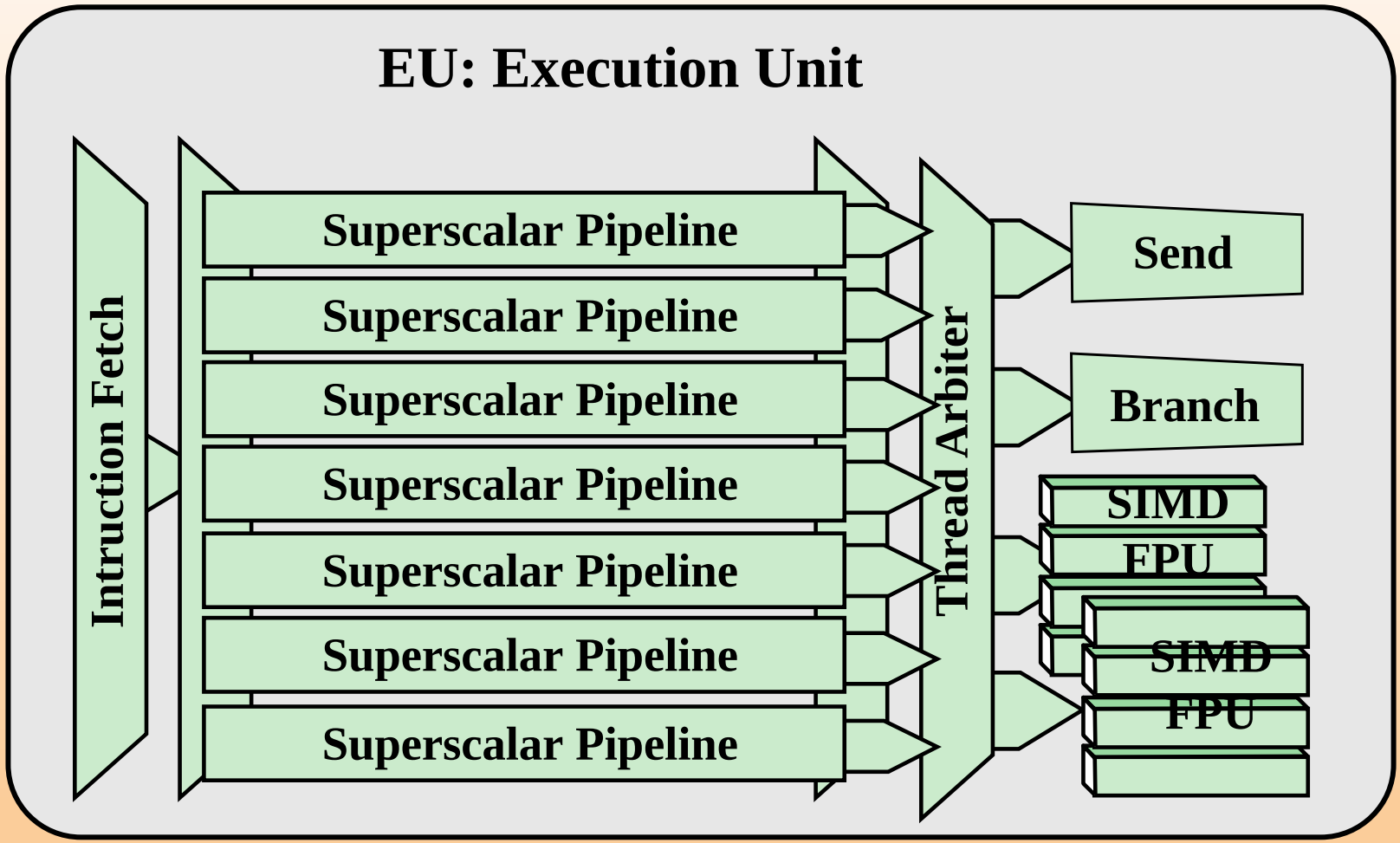


Figure 19.9 Intel Gen8 Execution Unit

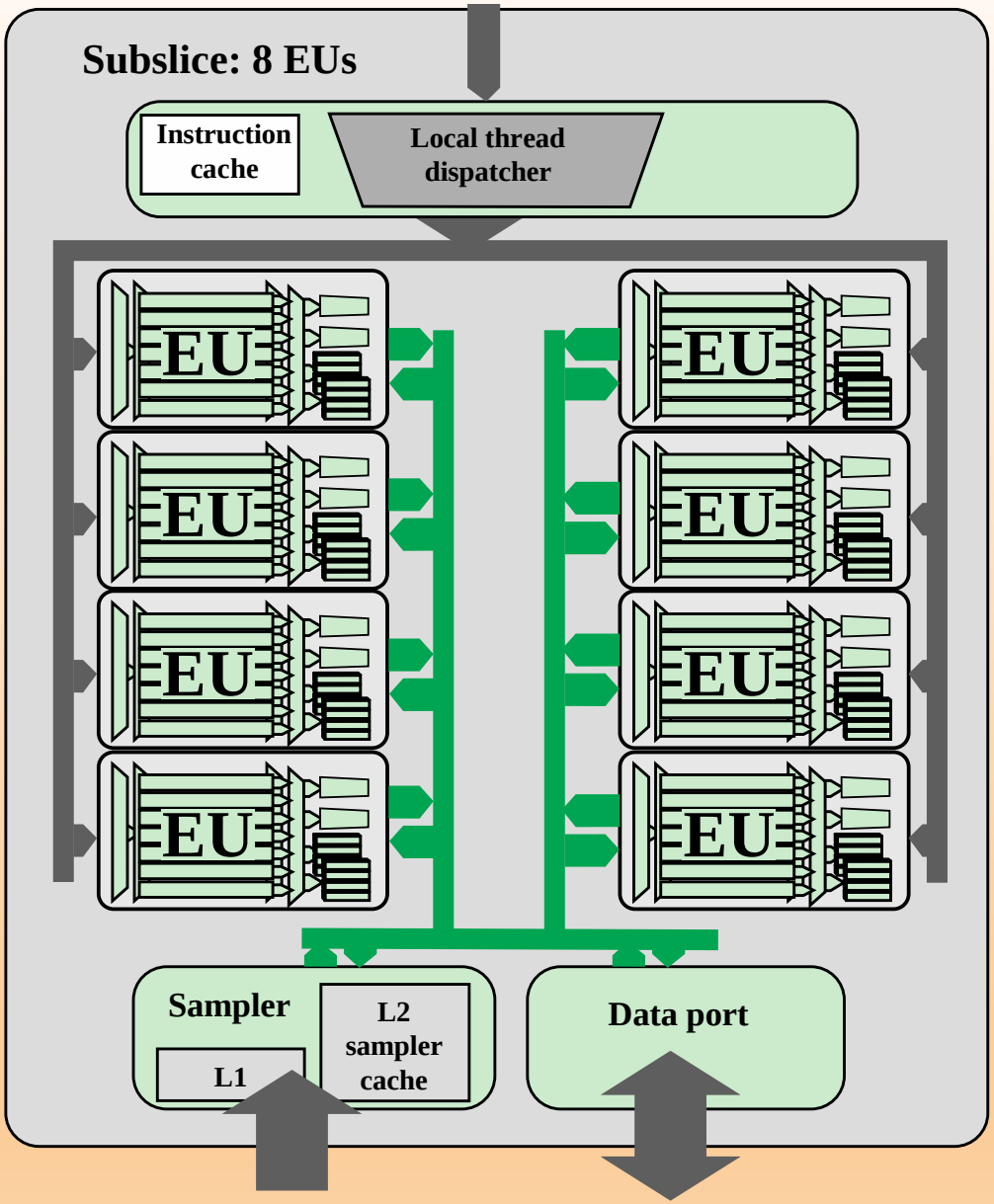


Figure 19.10 Intel Gen8 Subslice

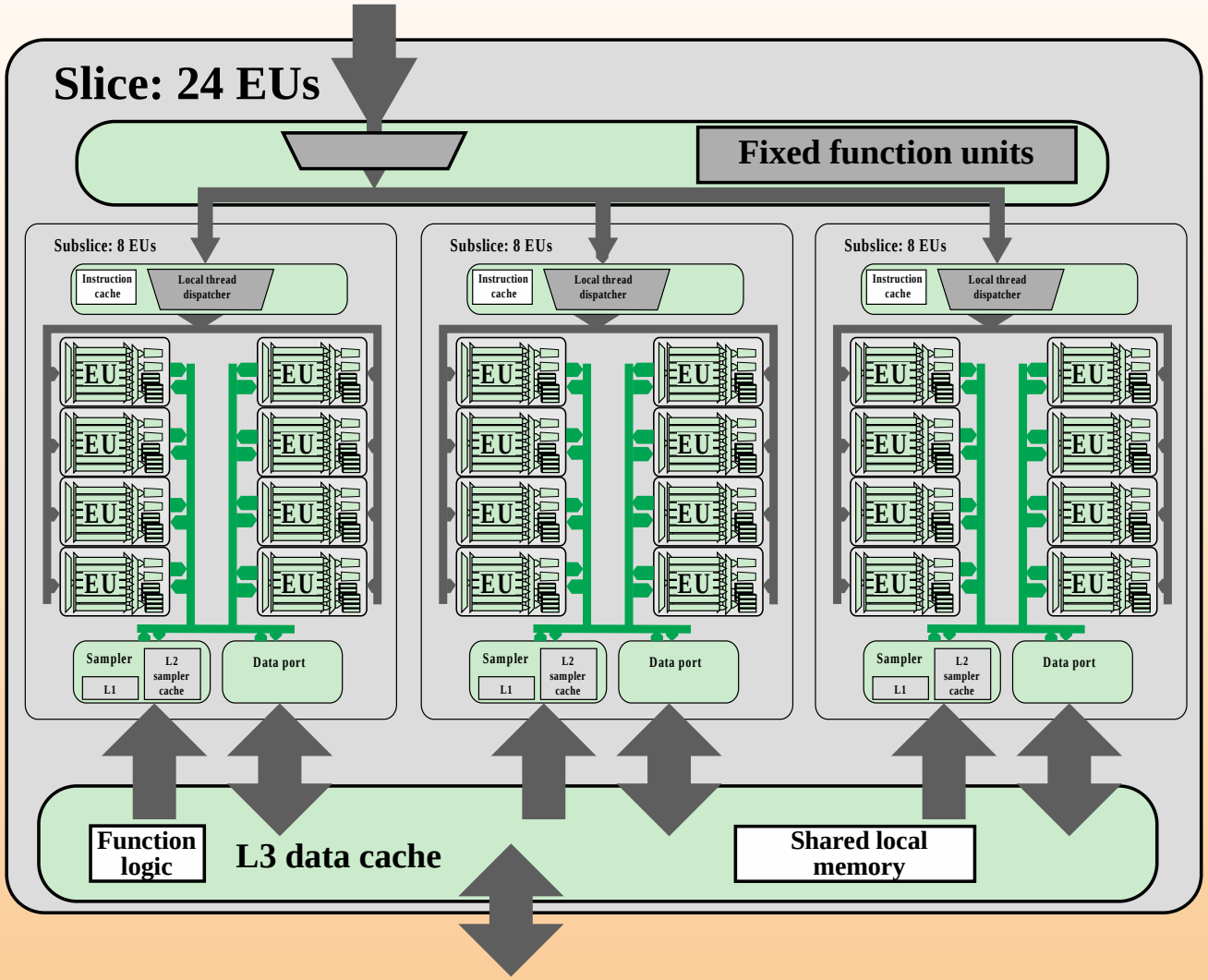


Figure 19.11 Intel Gen8 Slice

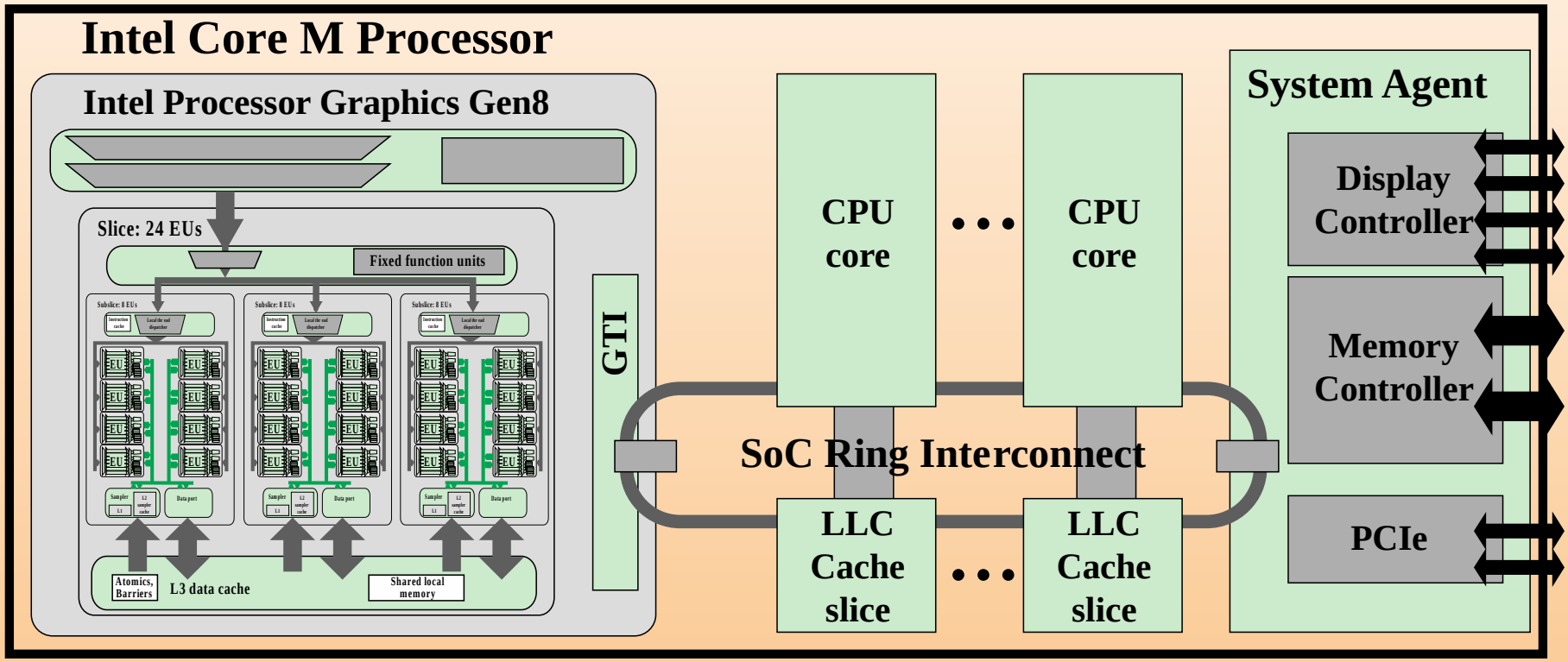


Figure 19.12 Intel Core M Processor SoC

Summary

Chapter 19

General-Purpose Graphic Processing Units

- CUDA basics
- GPU versus CPU
  - Basic differences between CPU and GPU architectures
  - Performance and performance per watt comparison
- Intel's Gen8 GPU
- GPU architecture overview
  - Baseline GPU architecture
  - Full chip layout
  - Streaming multiprocessor architecture details
  - Importance of knowing and programming to your memory types