Many-core Architectures

Sathish Vadhiyar

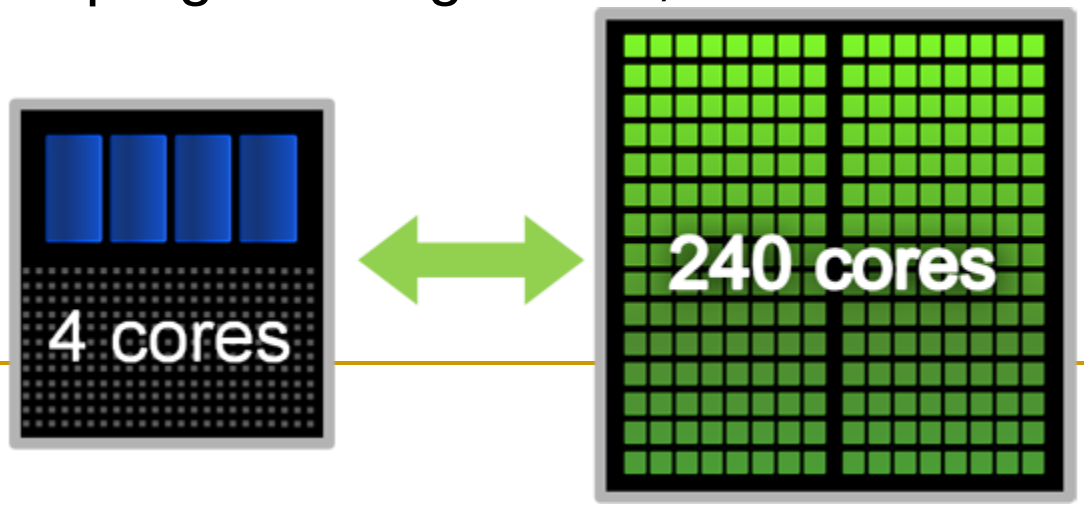
Introduction to Scalable Systems

Introduction

- Graphical Processing Unit
- A single CPU can consist of a max of 128 cores
- GPUs consist of a large number of light-weight cores
- Primarily proposed for graphics applications
  - Consists of stages with each stage consisting of independent computations

GPU and CPU, CUDA

- Typically GPU and CPU coexist in a heterogeneous setting
- “Less” computationally intensive part runs on CPU (coarse-grained parallelism), and more intensive parts run on GPU (fine-grained parallelism)
- NVIDIA’s GPU architecture is called CUDA (Compute Unified Device Architecture) architecture, accompanied by CUDA programming model, and CUDA C language



NVIDIA A100

- Has hierarchical architecture
- Graphic Processing Clusters (GPCs) – 7 Nos.
- Each GPC has:
  - 7 or 8 Nos. Texture Processing Clusters (TPCs). Hence a total of 64 TPCs
  - Up to 16 Nos. Streaming Multiprocessors (SMs). A total of 108 SMs

Figure 6. GA100 Full GPU with 128 SMs (A100 Tensor Core GPU has 108 SMs)

HBM2
HBM2
Memory Controller
Memory Controller
Memory Controller
GPC
GPC
TPC
SM
L2 Cache

NVIDIA A100 SM

- Each SM has:
  - 64 Nos. FP32 GPU cores. A total of 6912 GPU cores.
  - 4 tensor cores. Hence a total of 432 tensor cores. Together they deliver 1024 FP16/FP32 operations per clock
  - 192 KB of combined shared memory and L1 data cache

SM


SM
L1 Instruction Cache
L0 Instruction Cache
Warp Scheduler (32 thread/clk)
Dispatch Unit (32 thread/clk)
Register File (16,384 x 32-bit)
INT32
FP32
FP64
TENSOR CORE
LD/ST
SFU
192KB L1 Data Cache / Shared Memory
Tex

SM

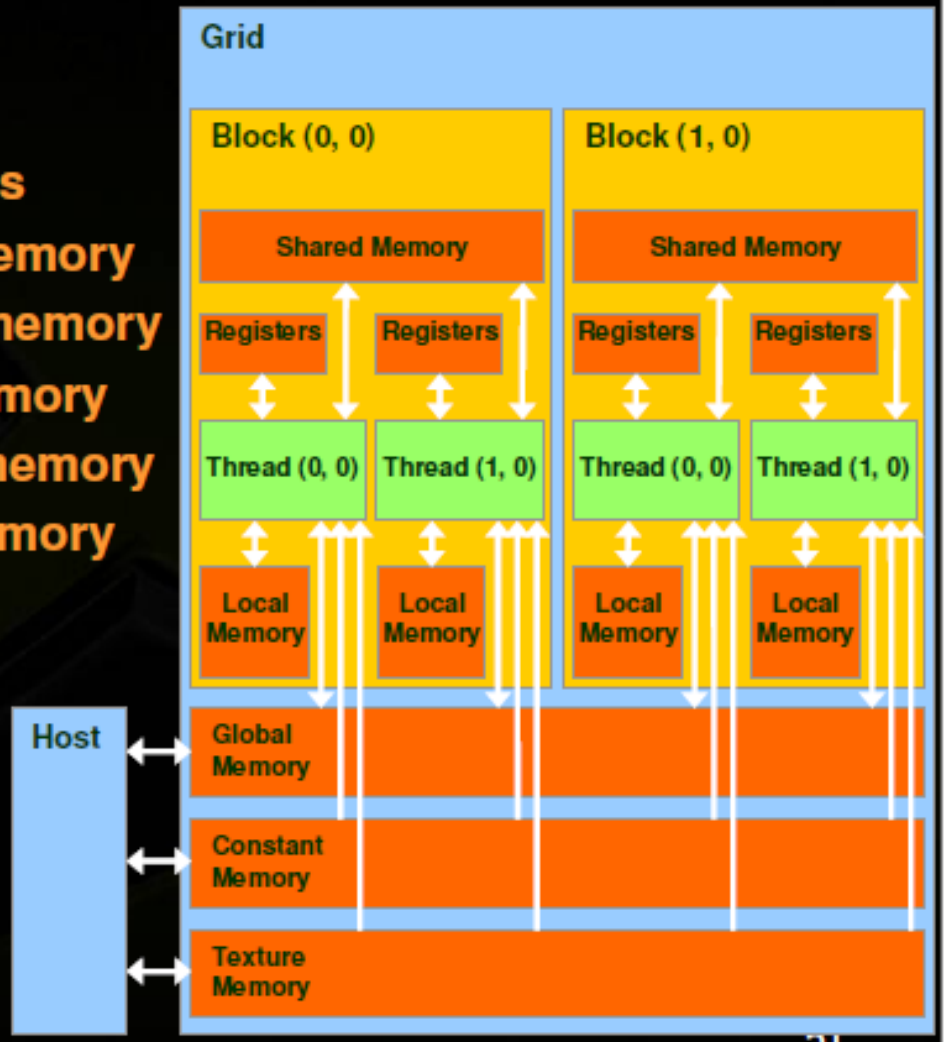

L0 Instruction Cache
Warp Scheduler (32 thread/clk)
Dispatch Unit (32 thread/clk)
Register File (16,384 x 32-bit)
INT32
FP32
FP64
TENSOR CORE
LD/ ST
SFU
192KB L1 Data Cache / Shared Memory
Tex

Tensor cores

- Specialized HPC compute cores for matrix operations for AI and HPC applications
- Perform matrix multiply and accumulate (MMA) operations

CUDA Memory Spaces

- Each thread can:
  - Read/write per-thread registers
  - Read/write per-thread local memory
  - Read/write per-block shared memory
  - Read/write per-grid global memory
  - Read only per-grid constant memory
  - Read only per-grid texture memory
- The host can read/write global, constant, and texture memory (stored in DRAM)



Memory Hierarchy

- Global or Device memory:
  - Can be accessed by all the threads executing in all the SMXs
  - Can be accessed by the CPU host
  - In A100s: 80 GB
- Shared memory:
  - In each SMX
  - Shared by all the threads of a thread block executing in a SMX
  - In A100s: 192 KB

Memory Hierarchy

- Registers
  - In each SMX
  - Used for storing the local data of the threads
  - Kepler k40: 64K registers in each SMX
  - A100: 64K registers in each SMX
- Constant and texture memory
  - Accessed by all the threads
  - Texture memory: used to improve performance of reads that exhibit spatial locality among the threads
- Latency of data access
  - Device memory: 200-400 clock cycles (about 300 ns)
  - Shared memory: 20-30 clock cycles (about 5 ns)

Differences with the CPU threads

- Context switching fast: The state of a thread (thread block) stored in shared memory and registers stay till execution completion
- Cache explicitly managed: User's program will have to explicitly bring the frequently accessed data from the device to the shared memory