Nvidia Ampere GA102

Deepti, Aditya, Jialin, Sowmya
Agenda

1. Introduction
2. Microarchitecture
3. Memory and Networking
4. Cores
   a. Ray Tracing Cores
   b. Tensor Cores
   c. Programmable Shading Cores
5. ADA vs Ampere vs Turing
Graphics Processing Unit (GPU)

- GPU renders images, video and 2D or 3D animations for display. A GPU performs quick math calculations and frees up the CPU to do other things.
- Integrated GPUs are located on the CPU
- Discrete GPUs live on their own card

Machine Learning  
3D Graphics Rendering  
Video Editing  
Cryptocurrency Mining

CPU vs GPU

- Low Compute Density
- Optimized for serial operations
- Shallow pipelines (<30 stages)
- Low latency tolerance
- Complex control logic

- High Compute Density
- Optimized for parallel operations
- Deep pipelines (100s)
- High latency tolerance
- High throughput
**Highlights**

The GA102 GPU is NVIDIA's largest Ampere GPU for the gaming & consumer segment. It is used in the top of the line GeForce RTX 3090 and GeForce RTX 3080 graphics cards.

It was released on September 1, 2020.
Microarchitecture

GigaThread Engine
Graphics Processing Clusters (GPCs)
Texture Processing Clusters (TPCs)
Streaming Multiprocessors (SMs)
Memory controllers
Graphics Processing Clusters

The GPC is the dominant high-level hardware block with all of the key graphics processing units.

Each GPC includes,

1. a dedicated Raster Engine
2. two raster operator partitions (each partition containing eight ROP units)
3. six TPCs
Graphics Processing Clusters

1. Raster Engine

2. Raster Operators - GPU's output is assembled into a bitmapped image ready for display.
   - Previously ROPs were tied to the memory controller and L2 cache.
   - GA102 boosts performance of raster operations by increasing the total number of ROPs, and eliminating throughput mismatches between the scan conversion frontend and raster operations backend.
Texture Processing Clusters (TPCs)

Each TPC includes,

1. two SMs
2. one PolyMorph Engine
Texture Processing Clusters (TPCs)

PolyMorph Engine - execution unit that handles geometry
Streaming Multiprocessors (SMs)

Each SM contains,

1. 128 CUDA Cores
2. 4 third-generation Tensor Cores
3. 4 Texture Units
4. 1 second-generation Ray Tracing Core
5. 128 KB of L1/Shared Memory, which can be configured for differing capacities
6. 256 KB Register File
7. L0 instruction cache, one warp scheduler, one dispatch unit
Streaming Multiprocessors (SMs)

These are general purpose processors with a low clock rate target and a small cache. The primary task of an SM is that it must execute several thread blocks in parallel. They support instruction-level parallelism but not branch prediction.

A warp is a set of 32 threads within a thread block such that all the threads in a warp execute the same instruction.
Memory hierarchy background

- **Global Memory** - Default main memory of GPU
- **Local Memory** - Abstraction on global memory with thread-level scope
- **Shared Memory** - On chip memory accessible to active threads in an SM
- **Constant Memory** - Predefined read-only space within global memory

Figure 1: GPU memory hierarchy
Memory Hierarchy for Ampere
GDDR6x Memory

- GDDR6 transmitted 2 bits per clock cycle, 1 on the rising edge and 1 on the falling edge
- GDDR6x transmits 2 bits each clock edge, that is 4 bits per clock cycle
- Encoded using 4 different voltage levels sent on each clock edge
- GDDR6x can effectively double bandwidth as compared to GDDR6 at a given operating frequency
Unified shared memory and L1 data cache

- Unified architecture for shared memory, L1 data cache and texture cache, similar to Turing architecture
- Reconfigurable to allocate more memory to L1 cache or shared memory based on workload
- GA10x GPUs have double the L1 cache bandwidth as compared to Turing (128 bytes/clock per SM vs 64 bytes/clock)
- GA102 GPU has 10752 KB of L1 cache as compared to 6912 KB in TU102 GPU
Network

- Third Generation NVLink
  - Four x4 links with each link providing 14.0625 GB/s bandwidth between 2 GPUs
  - Four links provides 56.25 GB/s bandwidth in each direction, i.e. 112.5 GB/s total bandwidth

- PCIe Gen 4
  - PCIe Gen 4 provides double the bandwidth as Gen 3 up to 16 GT/s
  - x16 PCIe 4.0 provides up to 64 GB/s peak bandwidth
Second-Generation Ray Tracing Engine in GA10x GPUs
Ampere Architecture Motion Blur Hardware Acceleration

TURING RT CORE

- BVH traversal
- Bounding box intersection
- Triangle intersection
- Return hit

GA10x RT CORE

- BVH traversal
- Bounding box intersection
- Interpolate triangle position (time)
- Triangle intersection
- Return hit

SM

Cast → Ray

RT Core

Perform all the calculations needed for BVH traversal and triangle intersection tests

RT Core

Return Hit/No hit → SM

MIMD Architecture
Table Ray Tracing Feature Comparison

<table>
<thead>
<tr>
<th></th>
<th>NVIDIA Turing Architecture (TU102 Full-Chip)</th>
<th>NVIDIA Ampere Architecture (GA102 Full-Chip)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dedicated RT Cores</td>
<td>Yes (72 RT Cores)</td>
<td>Yes (84 RT Cores)</td>
</tr>
<tr>
<td>Ray / Bounding Box Acceleration</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Ray / Triangle Acceleration</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Tree Traversal Acceleration</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Instance Transform Acceleration</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Concurrent RT and Shading</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Dedicated L1 Interface</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Ray / Triangle Intersection Test Culling Rate Speedup</td>
<td>1.0x</td>
<td>2.0x</td>
</tr>
<tr>
<td>Overall GPU RT Speedup</td>
<td>1.0x</td>
<td>2.0x</td>
</tr>
</tbody>
</table>
Async Compute: a feature that allows the GPU to perform compute and graphics workloads simultaneously.

2nd Generation RT Core

- Dedicated Hardware
- 2X Ray/Triangle Intersection
- Concurrent RT + Graphics
- Concurrent RT + Compute

Also allow scenarios such as a compute-based denoising algorithm to run concurrently with RT Core-based ray tracing work.

Second-Generation RT Core in GA10x GPUs
Leaving much SM processing power available for other workloads.

Second-Generation RT Core in GA10x GPUs

- Dedicated Hardware
- 2x Ray/Triangle Intersection
- Concurrent RT + Graphics
- Concurrent RT + Compute
Ampere Architecture Motion Blur Hardware Acceleration
WITHOUT MOTION BLUR

Rays (many_directions)

Solve triangle intersection

Output samples

Filtered output

WITH MOTION BLUR

Rays (many_directions, many_times)

Find time segment and solve for position = f(time)

Solve triangle intersection

Output samples

Filtered output

Interpolate Triangle Position unit

Generate triangles in the BVH in between existing triangle representations based on object motion

rays can intersect triangles at their expected positions in object space at the times specified by the ray timestamps

Allows accurate ray-traced motion blur rendering to occur up to eight times faster than the Turing GPU architecture.
Third-Generation Tensor Cores

Specialized execution units designed specifically for performing the tensor/matrix operations

Core Compute Function

Deep Learning
<table>
<thead>
<tr>
<th>GPU Architecture</th>
<th>TU102 SM (RTX 2080 Super)</th>
<th>GA10x SM (RTX 3080)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tensor Cores per SM</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>FP16 FMA operations per Tensor Core</td>
<td>64</td>
<td>Dense: 128</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sparse: 256</td>
</tr>
<tr>
<td>Total FP16 FMA operations per SM</td>
<td>512</td>
<td>Dense: 512</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sparse: 1024</td>
</tr>
</tbody>
</table>

Each Ampere architecture SM: Fewer Tensor Cores More powerful Tensor Core

Ampere Architecture Tensor Core vs Turing Tensor Core
Third-generation Tensor Core

- Accelerates more data types
- Includes a new Sparsity feature, delivering up to a 2x speedup for matrix multiplication
- Increases raw performance and brings new precision modes such as TF32 and BFloat16
CUDA programming model

- **CUDA blocks**—A collection or group of threads.
- **Shared memory**—Memory shared within a block among all threads.
- **Synchronization barriers**—Enable multiple threads to wait until all threads have reached a particular point of execution before any thread continues.

* Easy to scale
CUDA Cores

- Nvidia’s parallel processing platform: CUDA: Ampere GA102 has 10,752 CUDA cores
- CUDA Cores are the processing units inside a GPU → AMD’s Stream Processors.
- Effect on the Performance?
  - “More number of CUDA cores → More is the processing speed”?
  - For example: Number of CUDA cores for:
    - GTX 980: 2816
    - GTX 1080: 2560
Programmable Shading Cores

- Consists of NVIDIA CUDA Cores
- Shading units are small processors within the graphics card - responsible for processing aspects of the image.
- Have dedicated units for different types of operations in the rendering pipeline
- GA102 GPU incorporates 10752 CUDA Cores, 84 second-generation RT Cores, and 336 third-generation Tensor Cores
Pipeline Entities

More shading units a graphics card has → faster power allocation to process the workload
<table>
<thead>
<tr>
<th>Graphics Card</th>
<th>GeForce RTX 2080 Ti</th>
<th>GeForce RTX 3090 Ti</th>
<th>GeForce RTX 4090</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPU Codename</td>
<td>TU102</td>
<td>GA102</td>
<td>AD102</td>
</tr>
<tr>
<td>GPU Architecture</td>
<td>NVIDIA Turing</td>
<td>NVIDIA Ampere</td>
<td>NVIDIA Ada Lovelace</td>
</tr>
<tr>
<td>GPCs</td>
<td>6</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>TPCs</td>
<td>34</td>
<td>42</td>
<td>64</td>
</tr>
<tr>
<td>SMs</td>
<td>68</td>
<td>84</td>
<td>128</td>
</tr>
<tr>
<td>CUDA Cores / SM</td>
<td>64</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>CUDA Cores / GPU</td>
<td>4352</td>
<td>10752</td>
<td>16384</td>
</tr>
<tr>
<td>Tensor Cores / SM</td>
<td>8 (2nd Gen)</td>
<td>4 (3rd Gen)</td>
<td>4 (4th Gen)</td>
</tr>
<tr>
<td>Tensor Cores / GPU</td>
<td>544</td>
<td>336 (3rd Gen)</td>
<td>512 (4th Gen)</td>
</tr>
<tr>
<td>OFA TOPS³</td>
<td>N/A</td>
<td>126</td>
<td>305</td>
</tr>
<tr>
<td>RT Cores</td>
<td>68 (1st Gen)</td>
<td>84 (2nd Gen)</td>
<td>128 (3rd Gen)</td>
</tr>
<tr>
<td>GPU Boost Clock (MHz)</td>
<td>1635</td>
<td>1860</td>
<td>2520</td>
</tr>
</tbody>
</table>

ADA vs Ampere vs Turing
ADA vs Ampere vs Turing

- 2x GPCs (Versus Ampere)
- 50% More Cores (Versus Ampere)
- 50% More L1 Cache (Versus Ampere)
- 16x More L2 Cache (Versus Ampere)
- Double The ROPs (Versus Ampere)
- 4th Gen Tensor & 3rd Gen RT Cores
References

Backup Slides
GigaThread Engine

Scheduler and management cores that live inside of the GPU

Incorporates central graphics work scheduling, handing the work over to other subsections of the GigaThread Engine and finally communicating with the compute cores themselves.

This main core essentially runs an extremely basic "operating system".
Four cores (four fragments in parallel)
Instruction stream sharing

But ... many fragments should be able to share an instruction stream!

<diffuseShader>:
  sample r0, v4, t0, s0
  mul r3, v0, cb0[0]
  madd r3, v1, cb0[1], r3
  madd r3, v2, cb0[2], r3
  clmp r3, r3, l(0.0), l(1.0)
  mul o0, r0, r3
  mul o1, r1, r3
  mul o2, r2, r3
  mov o3, l(1.0)
Idea #2: Amortize cost/complexity of managing an instruction stream across many ALUs

SIMD processing
Modifying the shader

VEC8_diffuseShader:
VEC8_sample vec_r0, vec_v4, t0, vec_s0
VEC8_mul vec_r3, vec_v0, cb0[0]
VEC8_madd vec_r3, vec_v1, cb0[1], vec_r3
VEC8_madd vec_r3, vec_v2, cb0[2], vec_r3
VEC8_clmp vec_r3, vec_r3, l(0.0), 1(1.0)
VEC8_mul vec_o0, vec_r0, vec_r3
VEC8_mul vec_o1, vec_r1, vec_r3
VEC8_mul vec_o2, vec_r2, vec_r3
VEC8_mov o3, l(1.0)
128 in parallel

diagram showing vertices, primitives, and fragments in parallel processing.