NVIDIA Tesla K20X GPU Accelerator

Breton Minnehan, Beau Sattora
Overview

• Job of the GPU
• History
• What is the K20X
• GK110
• Benchmark Performance
Job of the GPU

- **Vertex Shader**
  - Applies transforms on each vertex
  - Applies color to vertices
  - Calculates normal at vertex

- **Fragment Shader**
  - Calculates fragment normal
  - Calculates fragment pixel colors and lighting
History of GPU

• Started as fixed-function graphics processors

• As they became increasingly programmable and more powerful, GPU's started being used in more scientific applications

• GPU has thousands of smaller, more efficient cores designed for parallel computations

• Required the use of graphics programming API's
  o OpenGL, Cg, DirectX

• GPUs designed for General Purpose computing
  o NVIDIA Tesla GPUs
History of GPU
The use of a GPU together with a CPU to accelerate general-purpose scientific and engineering applications

- Computationally-intensive portions of an application offloaded to GPU while the remaining code runs on the CPU
- Leverage the parallel processing capability in General Purpose Computing

- Researchers originally tried using OpenGL for General Purpose Computing
  - Had to map scientific calculations onto problems that could be represented by triangles and polygons
- NVIDIA created CUDA to facilitate use GPU for General Purpose Computing
CUDA

- Launched by NVIDIA in 2006
  - Currently on iteration 5
- Allows full programmability of the GPU
  - C/C++/C#/Fortran/Java
- Faster solutions to parallel processing
- Automatically parallelize loops
NVIDIA GPGPUs
What is the K20X?
# K20X Specifications

<table>
<thead>
<tr>
<th>GPU</th>
<th>GK110</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of CUDA Cores</td>
<td>2688</td>
</tr>
<tr>
<td>Processor Clock</td>
<td>732 MHz</td>
</tr>
<tr>
<td>Memory Clock</td>
<td>2.6 GHz</td>
</tr>
<tr>
<td>Memory Size (GDDR5)</td>
<td>6 GB</td>
</tr>
<tr>
<td>Peak Memory Bandwidth</td>
<td>250 GB/Second</td>
</tr>
<tr>
<td>Floating Point</td>
<td>3.95 Tflops (single)</td>
</tr>
<tr>
<td></td>
<td>1.31 Tflops (double)</td>
</tr>
</tbody>
</table>
Tesla K20X Block Diagram

- 6 GB GDDR5
- 24 pieces 64Mx16
- VBIOS 2Mbit ROM
- GK110
- Power Supply
- PCI Express Edge Connector
- Gen2 x16
- 12V/ 3V3
- 12V
- Aux Power 6-pin/8-pin
- GPU
- FB
- PEX
- GPIO
GK110

- Layout
- Energy Efficiency
- SMX
- Memory
- Dynamic Parallelism
- Hyper-Q
- Grid Management Unit
- NVIDIA GPUDirect
GK110 - Layout

<table>
<thead>
<tr>
<th>Manufacturing Process</th>
<th>28 nm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transistors</td>
<td>7.1 Billion</td>
</tr>
</tbody>
</table>
GK110 - Efficiency

- 3 times the performance per watt compared to Fermi
- 28nm process efficiency
- Sacrificed space for efficiency
  - Run SMXes at 1x clock vs previous 2x
    - More execution units for same performance
    - Requires less power for clocking logic
GK110 - SMX

- 15 SMXes per GK110
  - 14 on K20X
- Each SMX has
  - 192 CUDA cores (single)
  - 64 double units
  - 32 special function units
  - 32 load/store units
  - 16 texture filtering units
GK110 - SMX

- Warp Scheduler - four per SMX
  - Warp is group of 32 threads
  - Two Instruction dispatchers per WS
  - Double precision can be paired with other instructions
  - Register scoreboarding for long latency operations
  - Inter-warp scheduling
    - Removed math datapath data hazard detection hardware
      - Compiler encodes when data will be ready
      - Hardware only uses data known to be ready
  - Thread block level scheduling
GK110 - SMX

• New ISA encoding
  o Allows 255 registers per thread
• Shuffle
  o Transfer between threads in warp without shared memory
  o One instruction instead of two (store/load)
• Bindless textures
  o Texture state stored in object memory
  o Pass texture by handles
GK110 - Memory

- Six 64-bit memory controllers
- Cache flexibility
- Higher Bandwidth
- Fast DRAM I/O
- 384 bit GDDR5
- 5.2 GHz memory clock
- 65536 Registers/Multiprocessor
- L1 cache configurations (L1 cache/shared memory)
  - 16kb/48kb
  - 32kb/32kb
  - 48kb/16kb
GK110 - Memory

- 48kb read-only cache
  - accessible by SMX load operations
  - full-speed unaligned memory access
- L2 cache
  - 1536 KB
  - unified across SMXes
  - 2x bandwidth of fermi
- Memory Protection
  - Single-Error Correct Double-Error Detect ECC code
    - Register files, shared memory, L1 cache, L2 cache, DRAM
  - Single Error correction (parity check)
    - Read-Only Data Cache
GK110 - Dynamic Parallelism

- GPU can start processes based on results without going to the CPU
  - Kernels can start new kernels
  - Streams can start new streams
- Benefits nested loops, divide and conquer tasks, Adaptive grid computations, ease of programing
GK110 - Hyper-Q

- 32 independent CUDA Streams
  - Each stream has independent work queue
    - Hardware managed
  - Allows up to 32 CPU cores to schedule work simultaneously
- Maximize utilization of GPU
- Removes false dependencies for MPI processes
Hyper-Q

Easy speed-up for legacy MPI codes

FERMI
1 MPI Job

KEPLER
32 Concurrent MPI Jobs
GK110 - Grid Management Unit

- Flexible grid management and dispatch control system
  - Able to dispatch 32 active grids
  - Pause dispatch of new grids
  - Manages and prioritizes grid queue
    - Pending or suspended grids
    - Paused grids until dependencies complete
- Grids can be created through CUDA in SMX
  - Work can be queued without CPU involvement
• Multi-GPU communication
  o Same Server
  o Across network
• Directly exchange data
  o Independent of CPU
• Direct access from SSDs, NICs, and IB adapters
• Decreases MPI send/receive messages
• Uses less system memory
• Frees GPU DMA engines
## Benchmark Performance

### Tesla K20X Speed-Up over Sandy Bridge CPUs

<table>
<thead>
<tr>
<th>Category</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Material Science</td>
<td>WL-LSMS</td>
</tr>
<tr>
<td>Higher Ed</td>
<td>MATLAB (FFT)*</td>
</tr>
<tr>
<td>Physics</td>
<td>Chroma</td>
</tr>
<tr>
<td>Earth Science</td>
<td>SPECFEM3D</td>
</tr>
<tr>
<td>Molecular Dynamics</td>
<td>Amber</td>
</tr>
</tbody>
</table>

### System Config
- CPU results: Dual socket E5-2687w, 3.10 GHz
- GPU results: Dual socket E5-2687w + 2 Tesla K20X GPUs

*MATLAB results comparing one i7-2600K CPU vs with Tesla K20 GPU
Benchmark Performance

Double Precision Performance (DGEMM)

- Xeon E5-2687W: 0.17 Teraflop
- Tesla M2090 (Fermi): 0.43 Teraflop
- Tesla K20X: 1.22 Teraflop

Single Precision Performance (SGEMM)

- Xeon E5-2687W: 0.35 Teraflop
- Tesla M2090 (Fermi): 0.89 Teraflop
- Tesla K20X: 2.9 Teraflop
Benchmark Performance

Graph showing performance relative to 2x CPU:
- 2xCPU: 1.0x
- 2xCPU+ 1xM2090: 3.4x
- 2xCPU+ 2xM2090: 4.6x
- 2xCPU+ 1xK20X: 7.1x
- 2xCPU+ 2xK20X: 8.2x

CPU results: Dual socket E5-2687w; GPU results: Dual socket E5-2687w + 2 Tesla K20X GPUs, 64GB per node.
Benchmark Performance

3x3x1 Graphite

- Cray XK7-Tesla K20X
- Cray XK7-CPU

Compute Efficiency

# of Compute Nodes

0 500 1000 1500 2000 2500

0 5000000 10000000 15000000

4x
Real World Use

Titan: World’s #1 Open Science Supercomputer

18,688 Tesla K20X GPUs
27 Petaflops: 90% of Performance from GPUs