Overview

1. Fermi
2. Kepler
   a. SMX Architecture
   b. Memory Hierarchy
   c. Features
3. Improvements
4. Conclusion
5. Brief overlook into Maxwell
Fermi

- ~2010
- 40 nm TSMC (some mobile used 28 nm)
- 16 Streaming Multiprocessors
  - 32 CUDA cores
  - 16 load/store units
  - 4 Special Function Units (SFUs)
    - Sine, cosine, reciprocal, square root
- CUDA core
  - One Integer FPU + ALU (floating point)
Kepler

- ~2012 - 2014
- 28 nm technology TSMC
- On most GeForce 600, 700, and 800M series
- Designed with energy efficiency in mind
  - 2 Kepler cores uses 90% of one Fermi core
- Unified GPU clock
SMX Architecture

- 15 SMX (Next Generation Streaming Multiprocessor)
  - 192 single precision CUDA cores
  - 64 double precision units
  - 32 loads/store units
  - 32 SFUs
  - 16 texture units
  - 65,536 32-bit registers
  - 4 Warp Scheduler
Feature Overview

- Quad Warp Scheduler
- Shuffle Instructions
- Texture Improvements
- Atomic Operations
- Memory Hierarchy
- Dynamic Parallelism

- Hyper-Q
- Grid Management Unit
- GPU Direct
- NVENC
- General improvements/features
Quad Warp Scheduler

- A warp is 32 parallel threads
- Each SMX contains 4 warp scheduler
  - Each contains 2 instruction dispatch units allowing 2 independent instruction per cycle
  - Allows double precision operations alongside other operations (Fermi did not allow this)
Quad Warp Scheduler (Cont.)

- Removal of complex hardware that prevents data hazards
  - A multi-port register scoreboard
  - dependency checker block
- Used compiler to determine possible hazards
  - Simple hardware block provides this pre-determined information to the instruction
- Replaces power expensive hardware stage with simple hardware block
- Frees up die space
Shuffle Instructions

● Allows threads within a warp to share data
  ○ Previously needed separate store and load operations to pass data to shared memory
● Instead, move the thread so they can access another thread’s register
  ● Store and load is carried in a single step
● Reduces amount of shared memory needed
● 6% performance gain in FFT using shuffle
Indexed any-to-any

Shift right to n\text{th} neighbour

Shift left to n\text{th} neighbour

Butterfly (XOR) exchange
Texture Improvements

● Texture state is now saved in memory
  ○ Fermi used a fixed size binding table
    ▪ Assigned a entry when GPU needed to reference a texture
    ▪ Basically resulted in a 128 texture limit
● Obtained on demand
● Reduces CPU overhead and improves GPU access efficiency
Pre-Kepler texture binding model
128 simultaneous textures

Kepler bindless textures
over 1M simultaneous textures
Atomic Operations

- Read, write, modify operations performed without interruptions from other threads
- Important for parallel programming
- Added atomicMin, atomicMax, atomicAnd, atomicOr, atomicXor operations
- Native support for 64 bit Atomic ops
Memory Hierarchy

- Configurable 64KB shared memory
  - 16/32/48 KB L1 cache
  - 48/32/16 KB shared memory
- 48 KB read only cache
- 1536 KB L2 cache
- Protected by Single-Error Correct Double-Error Detect (SECDED) ECC code
- More bandwidth at each level compared to previous
<table>
<thead>
<tr>
<th></th>
<th>FERMI GF100</th>
<th>FERMI GF104</th>
<th>KEPLER GK104</th>
<th>KEPLER GK110</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute Capability</td>
<td>2.0</td>
<td>2.1</td>
<td>3.0</td>
<td>3.5</td>
</tr>
<tr>
<td>Threads / Warp</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>Max Warps / Multiprocessor</td>
<td>48</td>
<td>48</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>Max Threads / Multiprocessor</td>
<td>1536</td>
<td>1536</td>
<td>2048</td>
<td>2048</td>
</tr>
<tr>
<td>Max Thread Blocks / Multiprocessor</td>
<td>8</td>
<td>8</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>32-bit Registers / Multiprocessor</td>
<td>32768</td>
<td>32768</td>
<td>65536</td>
<td>65536</td>
</tr>
<tr>
<td>Max Registers / Thread</td>
<td>63</td>
<td>63</td>
<td>63</td>
<td>255</td>
</tr>
<tr>
<td>Max Threads / Thread Block</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>Shared Memory Size Configurations (bytes)</td>
<td>16K</td>
<td>16K</td>
<td>16K</td>
<td>16K</td>
</tr>
<tr>
<td></td>
<td>48K</td>
<td>48K</td>
<td>32K</td>
<td>32K</td>
</tr>
<tr>
<td></td>
<td>48K</td>
<td>48K</td>
<td>48K</td>
<td>48K</td>
</tr>
<tr>
<td>Max X Grid Dimension</td>
<td>$2^{16}$-1</td>
<td>$2^{16}$-1</td>
<td>$2^{32}$-1</td>
<td>$2^{32}$-1</td>
</tr>
<tr>
<td>Hyper-Q</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Dynamic Parallelism</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Dynamic Parallelism

- Allows the GPU to generate, synchronize, and control new work for itself
- Traditionally CPU issues work to the GPU
- Does not need to involve the CPU for new work
Hyper-Q

- Fermi had 16 concurrent work streams but all were multiplexed into 1 hardware work queue
- Created false dependencies
- Increased number of hardware managed connections (work queues) to 32
  - Each CUDA stream is internally managed and intra-stream dependencies are optimized
Hyper-Q working with CUDA Streams: in the Fermi model shown on the left, only (C, P) & (R, X) can run concurrently due to intra-stream dependencies caused by the single hardware work queue. The Kepler Hyper-Q model allows all streams to run concurrently using separate work queues.

<table>
<thead>
<tr>
<th>Fermi Model</th>
<th>Kepler Hyper-Q Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>STREAM 1</td>
<td>STREAM 2</td>
</tr>
<tr>
<td>A</td>
<td>P</td>
</tr>
<tr>
<td>B</td>
<td>Q</td>
</tr>
<tr>
<td>C</td>
<td>R</td>
</tr>
</tbody>
</table>

Single hardware work queue

Each stream receives its own work queue
Grid Management Unit (GMU)

- Grid = group of blocks
  - block = group of threads
- Manages and prioritizes grids that are to be passed into the CWD (CUDA Work Distributor) to be sent to the SMX units for execution
- Keeps the GPU efficiently utilized
**Fermi Workflow**

- Stream Queue
  - Ordered queues of grids
- One-way Flow
- Work Distributor
  - Tracks blocks issued from grids
  - 16 Active Grids

**Kepler Workflow**

- Stream Queues
  - Ordered queues of grids
- CUDA-Created Work
- Grid Management Unit
  - Pending & suspended grids
  - 1000's of pending grids
- Two-way link allows pausing dispatch
- Work Distributor
  - Actively dispatching grids
  - 32 Active Grids
- SM, SM, SM, SM
- SMX, SMX, SMX, SMX
GPU Direct

- Allows direct access to GPU memory from third party devices.
  - NICs, SSDs, etc
- Remote Direct Memory Access (RDMA)
- Does not need to involve the CPU
GPUDirect™
Direct Transfers between GPU and 3rd Party Devices

Server 1
CPU
GDDR5 Memory
GPU1
GDDR5 Memory
GPU2
PCI-e
Network Card

Server 2
CPU
GDDR5 Memory
GPU2
Network Card
GDDR5 Memory
GPU1
PCI-e
NVENC

- New hardware-based H.264 video encoder
- Previous models used CUDA cores
- 4 times faster while using less power
- Up to 4096x4096 encode
- 16 minute long 1080p, 30 fps video will take approximately 2 minutes
Improvements of Kepler

- Access up to 255 register per thread (compared to 63 for Fermi)
- Removal of shader clock
  - Fermi used a shader clock typically 2x the GPU clock
    - Achieves higher throughput
    - Uses more power
  - Runs off GPU clock
Cont.

- Up to 4 displays on one card
- 4k support
- GPU Boost
  - Dynamically scale GPU clock based on operating conditions
- Adaptive V-sync
  - Turns off v-sync when frames per sec drops below 60
  - Turns on v-sync when above 60 fps
Games Tear When **VSync** Off

![Graph showing FPS and Time with tearing when VSync off](image)

Adaptive **VSync**

![Graph showing FPS and Time with prevent tearing and reduce stuttering](image)

Tear Point #1 --->

Tear Point #2 --->
Cont.

- FXAA (Fast Approximate anti-aliasing)
  o Comparable sharpness to MSAA (Multisample anti-aliasing)
  o Uses less computation power
  o Smooths edges using pixels rather than the 3D model
8x MSAA

Average: 30.0 fps
Maximum: 31.8 fps

FXAA (High)

Average: 52.9 fps
Maximum: 86.8 fps
Cont.

- TXAA (Temporal anti-aliasing)
  - Mix of hardware anti-aliasing, custom CG film style AA resolve
  - high-quality resolve filter to work with the HDR-correct post processing pipeline
  - TXAA 1 offers visual quality on par with 8xMSAA with the performance hit of 2xMSAA, while TXAA 2 offers image quality that is superior to 8xMSAA, but with performance comparable to 4xMSAA.
3DMark11 - Performance

- 3DMark11 - Performance
  - GeForce GTX 680 (1066MHz / 6008 MHz) - 9,303
  - GeForce GTX 580 (772MHz / 4008MHz) - 7,713
  - Radeon HD 7970 (925MHz / 5500MHz) - 6,578

- GPU Score
  - GeForce GTX 680 (1066MHz / 6008 MHz) - 9,298
  - GeForce GTX 580 (772MHz / 4008MHz) - 7,406
  - Radeon HD 7970 (925MHz / 5500MHz) - 6,027

- Combined Score
  - GeForce GTX 680 (1066MHz / 6008 MHz) - 7,655
  - GeForce GTX 580 (772MHz / 4008MHz) - 7,010
Groundbreaking Power Efficiency

Perf / Watt

<table>
<thead>
<tr>
<th>Game</th>
<th>GTX 580</th>
<th>GTX 680</th>
</tr>
</thead>
<tbody>
<tr>
<td>Just Cause 2</td>
<td>1.6</td>
<td>1.5</td>
</tr>
<tr>
<td>Batman: AC</td>
<td>1.4</td>
<td>1.3</td>
</tr>
<tr>
<td>Civilization V</td>
<td>1.2</td>
<td>1.1</td>
</tr>
<tr>
<td>Crysis 2</td>
<td>1.0</td>
<td>0.9</td>
</tr>
<tr>
<td>Skyrim</td>
<td>0.8</td>
<td>0.7</td>
</tr>
<tr>
<td>Battlefield 3</td>
<td>1.6</td>
<td>1.5</td>
</tr>
<tr>
<td>Lost Planet 2</td>
<td>1.4</td>
<td>1.3</td>
</tr>
<tr>
<td>StarCraft II</td>
<td>1.2</td>
<td>1.1</td>
</tr>
<tr>
<td>Dirt 3</td>
<td>0.8</td>
<td>0.7</td>
</tr>
<tr>
<td>Dragon Age 2</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>F1 2011</td>
<td>0.4</td>
<td>0.3</td>
</tr>
<tr>
<td>Deus Ex</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>Shogun 2</td>
<td>0.8</td>
<td>0.7</td>
</tr>
</tbody>
</table>

All games run at 1920x1080. See Appendix for detail settings.
In Conclusion

- Improve Performance
- Improve energy efficiency
- “Many hands make light work”
Maxwell

- 28nm TSMC
- Early 2014 (ver 1)
- Late 2014 (ver 2 current version)
  - GTX 980, 970
- New SM architecture (SMM)
  - Efficiency - more active threads per SMM
- Larger shared memory
- Larger L2 cache
KEPLER

135% Performance/Core

MAXWELL 1st Generation

2x Performance/Watt
<table>
<thead>
<tr>
<th>GPU</th>
<th>GK107 (Kepler)</th>
<th>GM107 (Maxwell)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUDA Cores</td>
<td>384</td>
<td>640</td>
</tr>
<tr>
<td>Base Clock</td>
<td>1058 MHz</td>
<td>1020 MHz</td>
</tr>
<tr>
<td>GPU Boost Clock</td>
<td>N/A</td>
<td>1085 MHz</td>
</tr>
<tr>
<td>GFLOPs</td>
<td>812.5</td>
<td>1305.6</td>
</tr>
<tr>
<td>Texture Units</td>
<td>32</td>
<td>40</td>
</tr>
<tr>
<td>Texel fill-rate</td>
<td>33.9 GigaTexels/sec</td>
<td>40.8 GigaTexels/sec</td>
</tr>
<tr>
<td>Memory Clock</td>
<td>5000 MHz</td>
<td>5400 MHz</td>
</tr>
<tr>
<td>Memory Bandwidth</td>
<td>80 GB/sec</td>
<td>86.4 GB/sec</td>
</tr>
<tr>
<td>ROPs</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>L2 Cache Size</td>
<td>256KB</td>
<td>2048KB</td>
</tr>
<tr>
<td>TDP</td>
<td>64W</td>
<td>60W</td>
</tr>
<tr>
<td>Transistors</td>
<td>1.3 Billion</td>
<td>1.87 Billion</td>
</tr>
<tr>
<td>Die Size</td>
<td>118 mm²</td>
<td>148 mm²</td>
</tr>
<tr>
<td>Manufacturing Process</td>
<td>28-nm</td>
<td>28-nm</td>
</tr>
</tbody>
</table>
Relative GPU Performance

Using 3DMark Vantage (Performance Preset)
GeForce GTX 980 2560x1600 Game Performance
Max Settings, High AA, 3960X, 8GB DDR3 RAM, Win 8.1 x64

<table>
<thead>
<tr>
<th>Game</th>
<th>Average Frames Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Watch Dogs</td>
<td>20</td>
</tr>
<tr>
<td>Hitman Absolution</td>
<td>25</td>
</tr>
<tr>
<td>Assassin's Creed IV</td>
<td>30</td>
</tr>
<tr>
<td>Total War: Rome 2</td>
<td>35</td>
</tr>
<tr>
<td>Tomb Raider</td>
<td>40</td>
</tr>
<tr>
<td>Sleeping Dogs</td>
<td>45</td>
</tr>
<tr>
<td>Far Cry 3</td>
<td>50</td>
</tr>
<tr>
<td>Thief</td>
<td>55</td>
</tr>
<tr>
<td>Battlefield 4</td>
<td>60</td>
</tr>
<tr>
<td>Call of Duty: Ghosts</td>
<td>65</td>
</tr>
<tr>
<td>GRID Autosport</td>
<td>70</td>
</tr>
<tr>
<td>BioShock Infinite</td>
<td>75</td>
</tr>
</tbody>
</table>
GEFORCE GTX 980 PERFORMANCE PER WATT

[Bar chart showing performance per watt for various games and benchmarks using GTX 680 and GTX 980 graphics cards.]
Questions?