Accelerating Cloud Graphics

Franck DIARD, Ph. D.

SW Architect
Distinguished Engineer, NVIDIA
Agenda

- 30 minute talk
- 10 minute demo
- 10 minute Q&A
GeForce® GRID

- Lower Latency
- Higher Density
- Higher Quality
Scope

- GeForce GRID
  - Coherent set of technologies
  - Moving GPUs into the Cloud, providing scalability
  - Overcoming density and cost challenges

- Hardware
  - GPU architecture / System integration

- Software
  - APIs, SDK, SW environment
  - Virtualization
  - Clients
Kepler, the First Cloud GPU

- High performance per watt
- Integrated hardware encoder
- Low-latency frame buffer reads
- GPU Virtualization
- 28nm
What for?

- Streaming anything from Cloud GPUs
  - Gaming
  - Enterprise workstation/VDI
  - Consumer desktop

- Mobile clients
  - Tegra3
  - Low power playback
  - Convenience
GeForce GRID Latency

CLIENT
- Render
- Decode
- Kybd/Mse

SERVER
- Render
- Capture
- Encode

Network
- GeForce GRID
  - <16ms
- Network
  - 30 ms
  - 2 Frames
- GeForce GRID
  - <30 ms

GPUS TECHNOLOGY CONFERENCE
## Server Optimized GPU

<table>
<thead>
<tr>
<th>GPUs</th>
<th>CUDA Cores</th>
<th>Memory Size</th>
<th>Memory Perf</th>
<th>Shader Perf</th>
<th>TDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dual</td>
<td>3,072</td>
<td>8GB</td>
<td>320 GB/sec</td>
<td>4.7 TFLOPS</td>
<td>250W</td>
</tr>
</tbody>
</table>
Server Grade Solution

- Passive cooling
- High Quality Components
**Power - Density, Savings**

- **Power management library**
  - Per GPU Power capping

First Generation Cloud: 1 GPU / Server, 1 Game Stream / Server, 150W / Game Stream

GeForce GRID Game Servers: 4 GPUs / Server, 4 Game Streams / Server, 75W / Game Stream
GeForce Grid Software

- **SDK**
  - 1 Header file, 1 DLL
  - Set of code samples, documentation

- **Server Side**
  - Accelerated frame grabbing
  - Video compression
  - Virtualization

- **Client Side**
  - PC low latency HW decode/display
  - Tegra low latency decode/display
GPU Virtualization

- Increase density, cutting cost

- NVMOS
  - Deployment, isolation
  - nvidia.com NVIDIA driver in VM at bare metal speed

- SDK: API Shimming
  - Injection of application
  - Inserting in band encoding calls
  - Allows n games to run on GPU
GeForce Grid: Virtualization

REMOTE GRAPHICS

NVENC
Low latency Encoder

Low Latency
Frame Buffer Capture

Low Latency
Render Target Capture

NVMOS
Platform Virtualization
Dedicated GPU

Windows 7+
Game

Windows 7+
Game

Game

Game

Windows 7+
Ad-hoc API

Shimming
DirectX

GPU

GPU

GPU

GPU
Frame Grabbing

- **Low latency**
  - Using async units in GPU, 0 CPU cycles

- **Convenient**
  - Minimal API, fast integration in existing stacks

- **Flexible API**
  - To HW H.264 encoder fastpath
  - To system memory for CPU codecs
  - To CUDA buffer for specialized codecs
Whole Display Grabbing

- Asynchronous Windows7 display grabber
- Orthogonal to all GFX stacks (gdi, dx9, dx10, OGL)
- Windows7 head, desktop games, flash games
- Standard Windows API
  - does not grab all cases
  - incurs a severe performance hit
Whole Display Grabbing

- HW overlay, HW mouse, Aero on, off, transitions

- Tear-free, all DMA, not vsync’d, format conversion, scaling

- Performance:
  - 4 ms to H.264 encoder, bits written back in system memory (720p)
  - 2 ms API call to system memory
  - 0.1 ms to CUDA
Render Target Grabbing

- SDK to use with API shimming
- Render target read back: Dx9, Dx10, Dx11 (OGL planned)
  - format conversion, scaling
- In band with GFX API: Present() call
  - Page locked sysmem
  - H.264 interface
  - CUDA interoperability
- Asynchronous Event Signaling
  - Not blocking main render loop
  - CPU friendly, interrupt driven
H.264 HW Encoding

- Completely separate GPU unit: <2 watts
- PSNR
  - comparable to x264
- up to 32 encoding contexts
  - 4 HD streams @60fps
- High Profile
  - 720p: 4 ms
  - 1080p: 8 ms
H.264 Encoder Features

- Constrained VBV buffer size
  - network packet framing for real time delivery
- CBR, VBR, Min QP
- CUDA Interoperability
- I-frame on-demand
- Max frame/slice size Capping
- Reference picture invalidation logic API for packet loss
- 4Kx4K support
- Stereo MVC Encoding
Client Side

- Client side is important
  - easy to ruin the user experience

- Generic, CPU based plugins
  - Slow decode and multi-frame buffering increase latency
  - Slow render of decoded output
  - CPU cycles burn a lot of power

- GeForce GRID for client: low latency and low power
  - GPU offload for decode and fast render
  - CPU just drives the IP stack and feeds GPU hardware
Client Side PC SDK

- SDK for:
  - bits in, frame out on the screen, lean and mean
  - feeding from system memory buffer

- HW decode on all nvidia GPUs: Windows, Linux
  - 60 FPS HD on common NVIDIA GPUs

- CUDA/DX/OGL interoperability
  - Gamma correction
  - Titling/HUD
Client Side Tegra3 SDK

- **SDK:**
  - HoneyComb/ICS and up, native
  - No added frame latency
  - Bypass of OS traditional stack, this is not streaming

- Decoder: 8ms 720p

- Tear free display

- 720p / 60 FPS

- 1080p / 30 FPS
Recorded Demo

- Gaming with Tegra3 over WIFI from local server
Recorded Demo

- Win8 with Tegra3 over WIFI from local server
Live Demos

- **Server**
  - Corei7 2.6 Ghz
  - @NVIDIA Headquarters, Santa Clara (10 miles away)
  - Bare metal Win7 32
  - Kepler GeForce GRID edition
    - 4GB FB, 1500 cores
BF3

- Tegra Transformer Prime
  - USB/Ethernet
- 720p
- 5Mbps
- 30 fps
- HW H.264 Encoding, high profile, no B frames
Desktop Remoting

- Aero On
- 1080p
- 5 Mbps

- Google SketchUp
- Flash/Web gaming
- Video playback
  - HW overlay on server
High End Gaming

- DX10
- 8 Mbps
- 1080p
Thanks To GeForce GRID Partners

- Gaikai
- Ubitus
- Playcast
- G-Cluster
- Otoy
- ...

Q&A

- Questions?

- Main contact at NVIDIA
  - Jon Barad jbarad@nvidia.com

Thanks!