



Quadro Kepler for Developers Theater @ SIGGRAPH 2012

Ian Williams – Director Applied Engineer, NVIDIA

Shalini Venkataraman – Senior Applied Engineer, NVIDIA

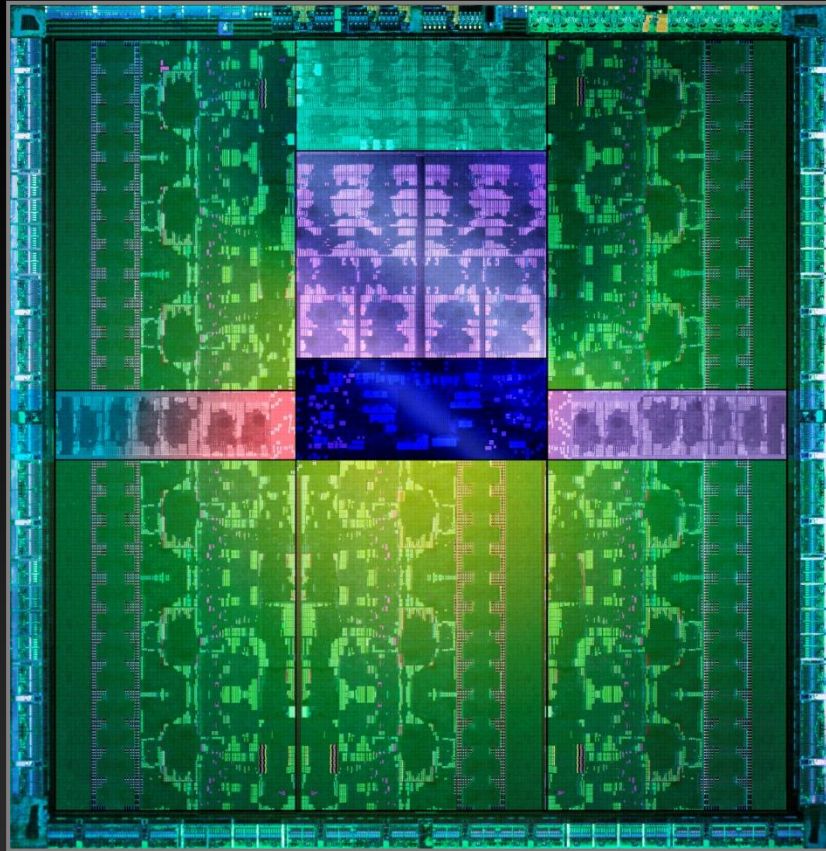


Quadro Kepler for Developers



Topics:

- **Kepler Architecture**
- **Maximus**
- **OpenGL 4.3**
- **WMI**
- **Developer Tools**
- **Summary**



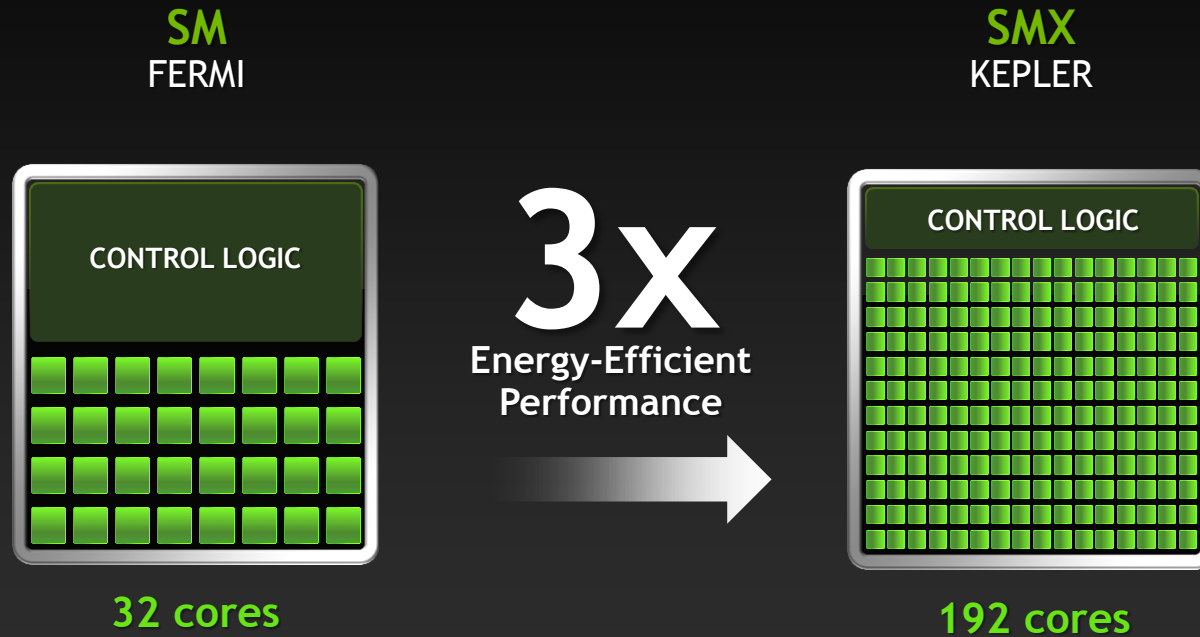
KEPLER

Architectural Features

- SMX
- PCIe Gen 3
- Texture improvements
- FXAA/TXAA
- Display Features
- Dynamic Parallelism

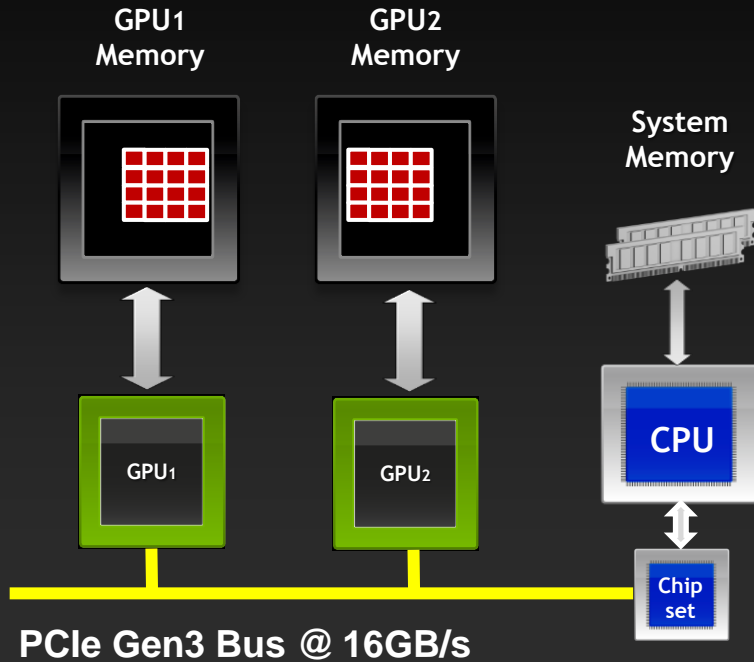
SMX

Delivers more processing performance and efficiency through an innovative streaming multiprocessor design that allows a greater percentage of space to be applied to processing cores versus control logic



PCIe Gen3

Next generation bus interconnect improves GPU to system to GPU data throughput for applications that require large data movement between system resources

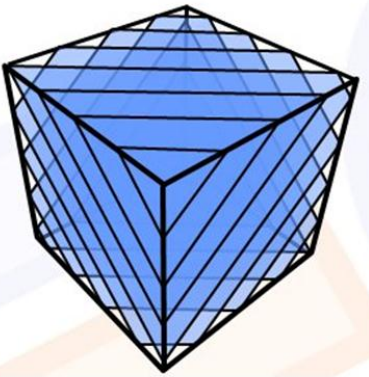


2x
Bandwidth
vs.
PCIe Gen2

Kepler Texture Improvements

● Increased Texture Units

Texture Slicing



Object-Order
CPU - Plane-box intersection
GPU - 3d texture lookup + blending

Raycasting (CUDA & GLSL)

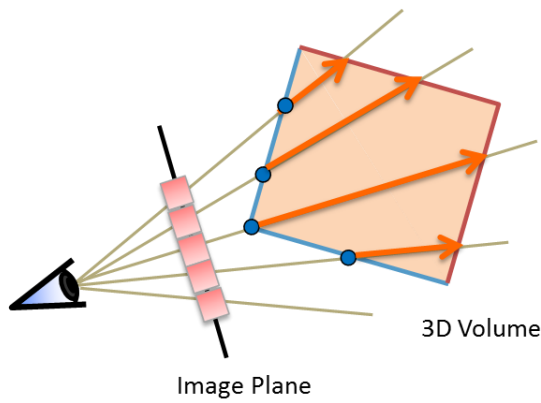
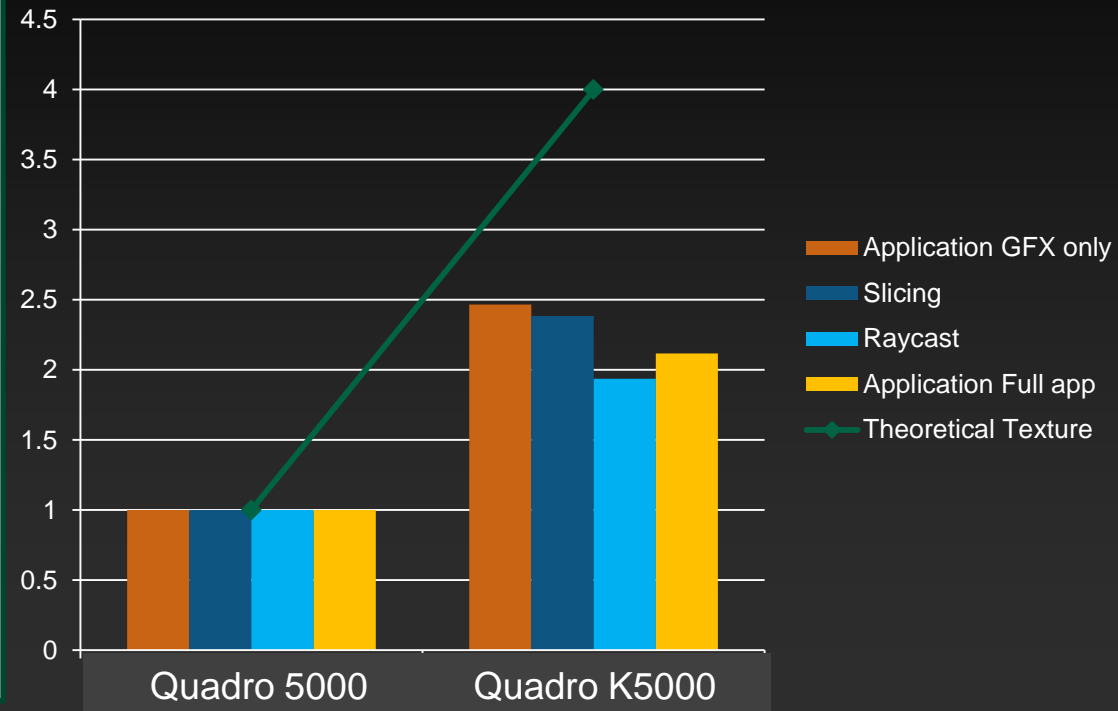


Image-Order
CPU - invokes fragment shader or cuda kernel
GPU - All the heavy lifting!
Raycasting + texture lookup + blending

Quadro K5000

Texture (3D) Scaling



Kepler Texture Improvements cont.

- Deep 3D Textures
 - Z >> X & Y
- Extension GL_NVX_deep_textures
- GL_MAX_3D_TEXTURE_SIZE will still be 4k.
- Introduce two or three new glGet() queries for:
 - GL_MAX_DEEP_3D_TEXTURE_WIDTH = 2k
 - GL_MAX_DEEP_3D_TEXTURE_HEIGHT = 2k
 - GL_MAX_DEEP_3D_TEXTURE_DEPTH = 16k

Kepler Texture Improvements cont.

- Increased Texture Sizes:

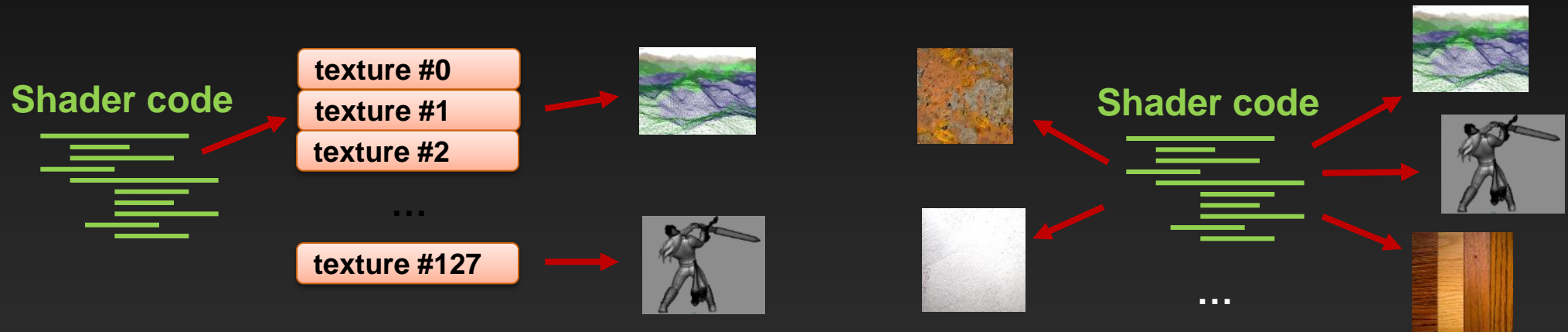
Architecture	OpenGL/ GeForce	OpenGL/ Tesla	OpenGL/ Quadro	Cuda/ GeForce	Cuda/ Tesla	Cuda/ Quadro
Fermi	512 ³	512 ³	2k ³	2k ³	2k ³	2k ³
Kepler	2k ³	2k ³	4k ³ , Deep	2k ³	4k ³ , Deep	4k ³ , Deep

Bindless Graphics

- **Problem:** Binding to different objects (textures, buffers) takes a lot of validation time in driver
 - And applications are limited to a small palette of bound buffers and textures
 - Approach of OpenGL, but also Direct3D
- **Solution:** Exposes GPU virtual addresses
 - Let shaders and vertex puller access buffer and texture memory by its virtual address!

Kepler - Bindless Textures

- Enormous increase in the number of unique textures available to shaders at run-time
- More different materials and richer texture detail in a scene



Pre-Kepler texture binding model

*Kepler bindless textures
over 1 million unique textures*

Kepler - Bindless Textures

Pre-Kepler texture binding model

CPU

Load texture A
Load texture B
Load texture C
Bind texture A to slot I
Bind texture B to slot J
Draw()



GPU

Read from texture at slot I
Read from texture at slot J

CPU

Bind texture C to slot K
Draw()



GPU

Read from texture at slot K

Kepler bindless textures

CPU

Load textures A, B, C
Draw()

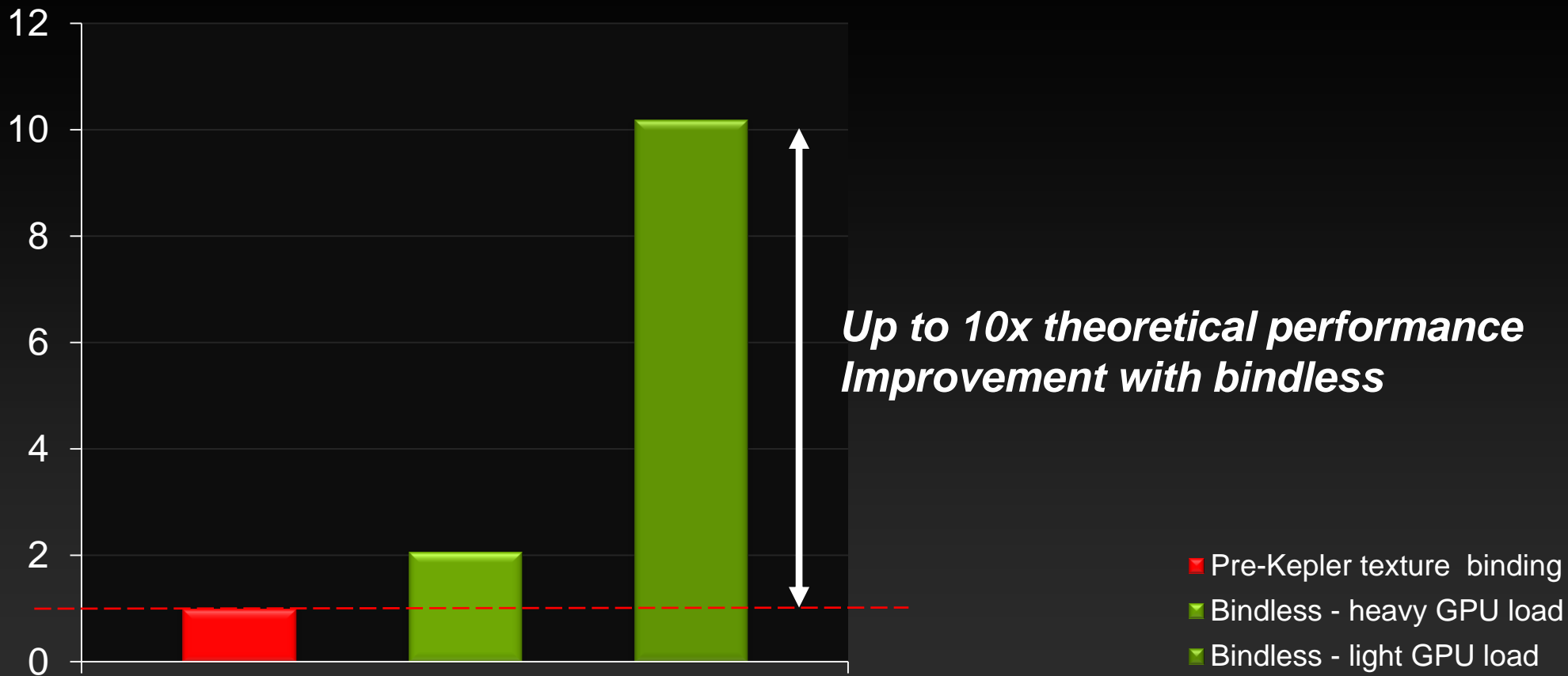


GPU

Read from texture A
Read from texture B
Read from texture C

Bindless model reduces CPU overhead and improves GPU access efficiency

Bindless performance benefit



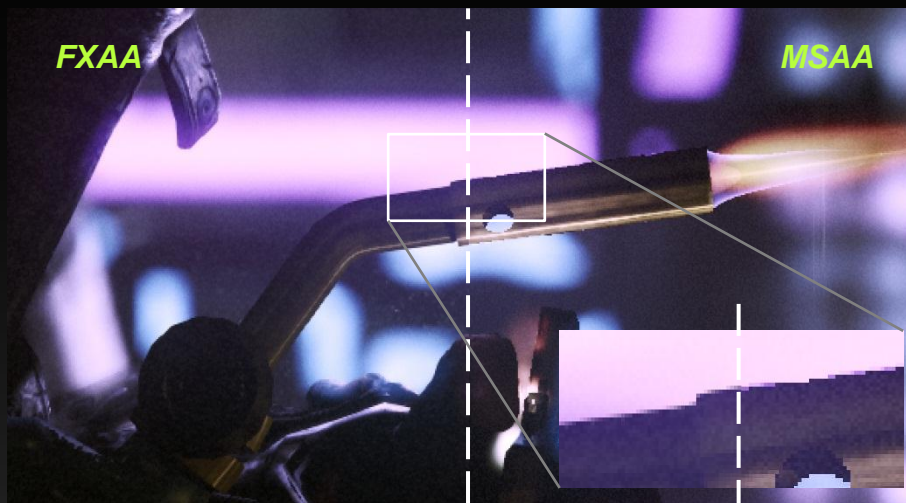
Numbers obtained with a directed test

FXAA / TXAA

- FXAA user configurable
- Allows higher quality filtering without requiring sub-sample storage
- Higher quality for the lower memory footprint

- TXAA is a developer feature
- Currently DX
- Temporal Filtering in addition to spatial filtering
- Higher quality

FXAA / TXAA



QUALITY



PERFORMANCE

Kepler Display Capabilities

- 4 Displays per GPU*
- Combines with:
 - Premium Mosaic mode
 - Warp and Blend API
 - GPU Direct for video (low latency I/O)



* Win XP limited to 2 displays

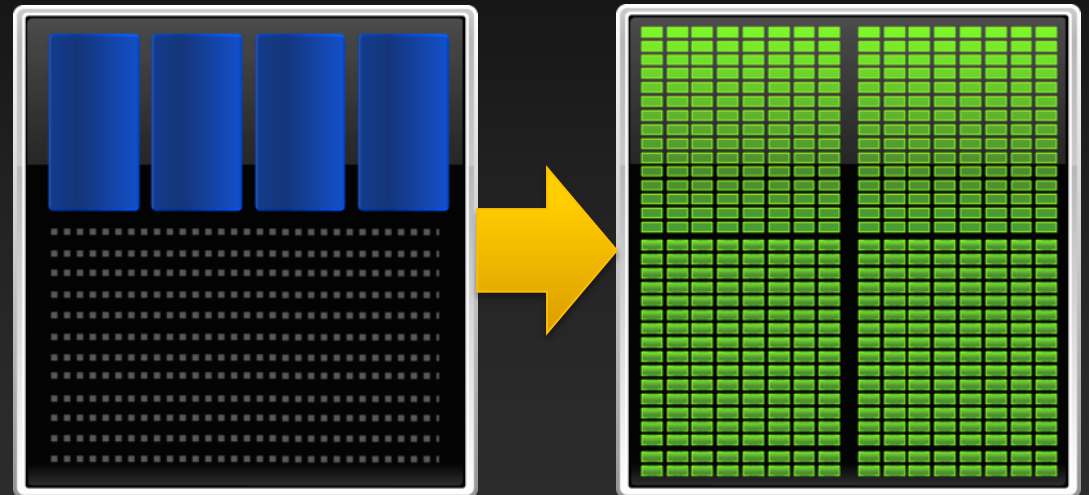
New Options for Collaborative Displays



Including life Size Retina Displays

Development Trends...

- More non-graphics developers are moving code to the GPU
- More graphics and professional application developers are mixing Compute and OpenGL
 - Computational graphics
 - Finite element analysis
 - Volume rendering
 - Tessellation
 - Ray tracing
 - Simulation
 - ...



Dynamic Parallelism

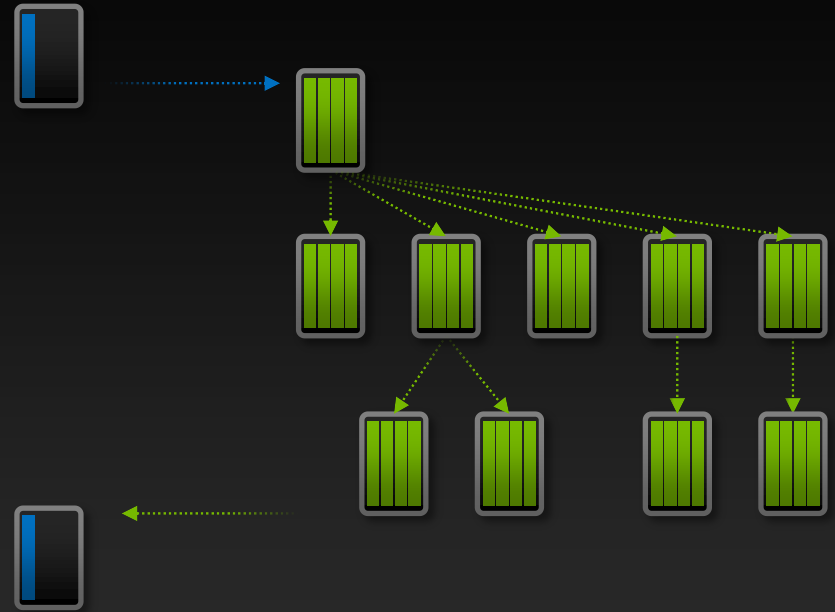
CPU

Fermi GPU



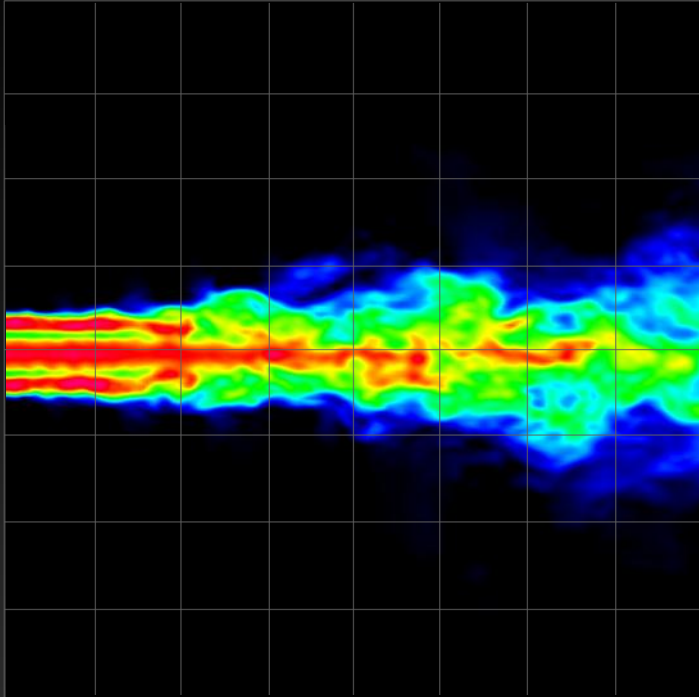
CPU

Kepler GPU



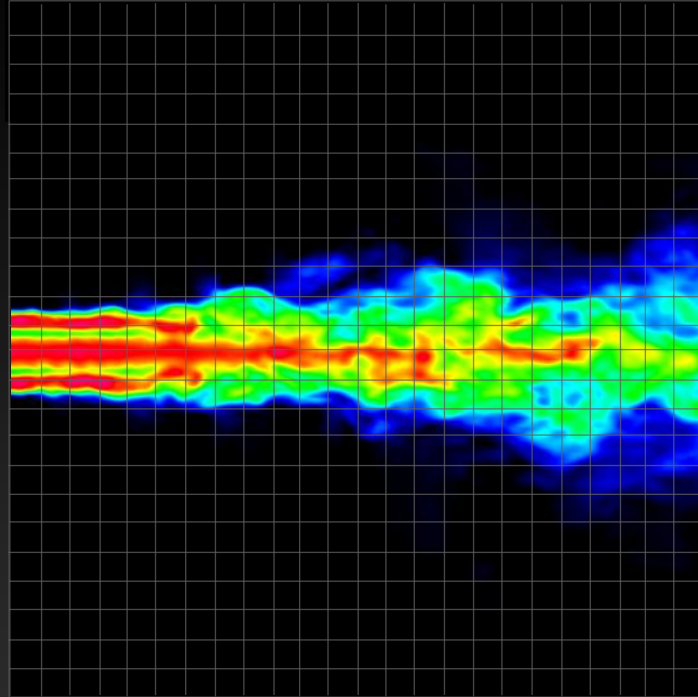
Dynamic Work Generation

Coarse grid



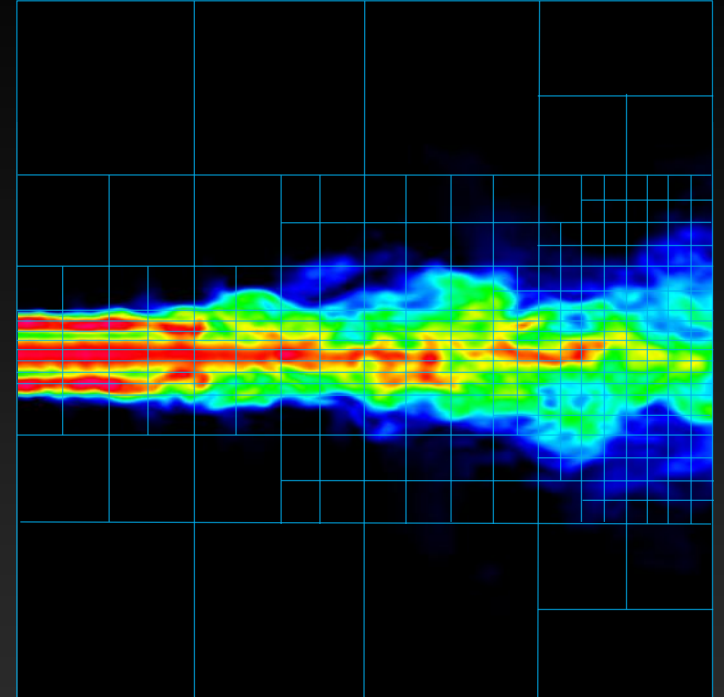
Higher Performance
Lower Accuracy

Fine grid



Lower Performance
Higher Accuracy

Dynamic grid

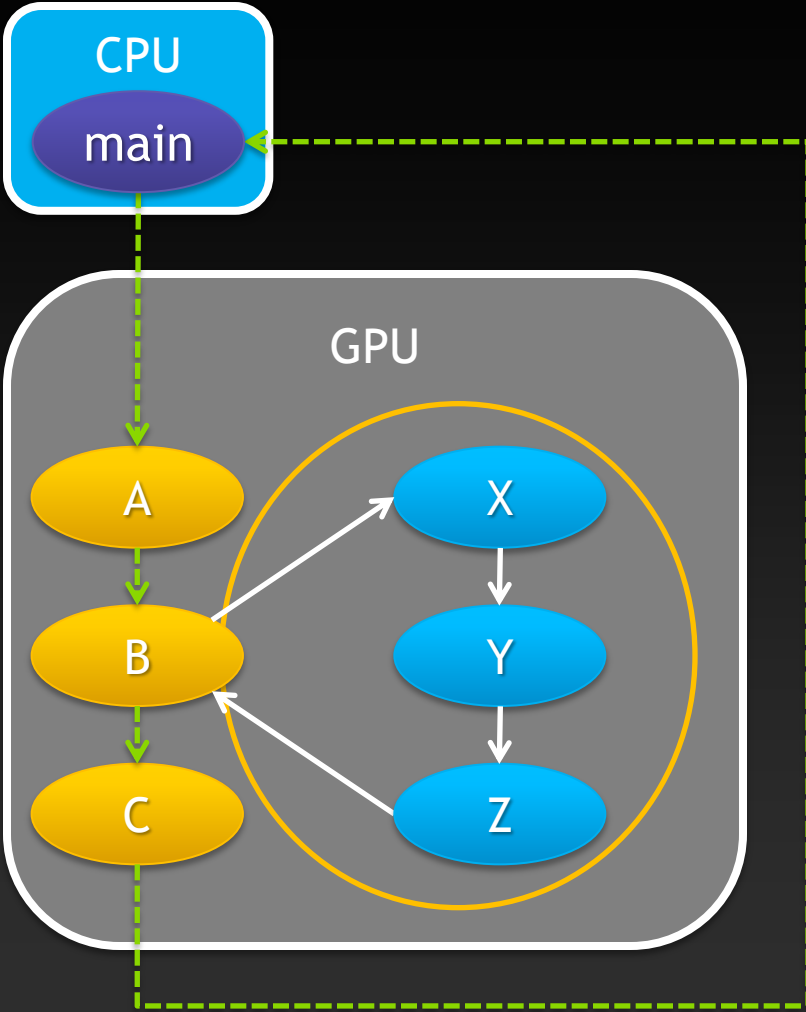


Target performance where
accuracy is required

Familiar Syntax and Programming Model

```
int main() {  
    float *data;  
    setup(data);  
  
    A <<< ... >>> (data);  
    B <<< ... >>> (data);  
    C <<< ... >>> (data);  
  
    cudaDeviceSynchronize();  
    return 0;  
}
```

```
__global__ void B(float *data)  
{  
    do_stuff(data);  
  
    X <<< ... >>> (data);  
    Y <<< ... >>> (data);  
    Z <<< ... >>> (data);  
    cudaDeviceSynchronize();  
  
    do_more_stuff(data);  
}
```



Maximus for Developer

- **Visualization + Compute**
 - Eg 4D time varying
- **Best characterized by:**
 - What features did you leave on the table because you didn't have enough compute power?
 - What would you have implemented if you had a dedicated teraflop of compute power?

Programming for Compute and Graphics

- Check out the Tech Talk:

9:00 AM - 10:30 AM GPU Programming for High-Performance Graphics
Workstation Applications

Alina Alt, Senior Applied Engineer, NVIDIA

Wil Braithwaite, Senior Applied Engineer, NVIDIA

Shalini Venkataraman, Senior Applied Engineer, NVIDIA

OpenGL 4.3

- OpenGL 4.3 is truly super set of DX 11
- Introduces the OpenGL GLSL Compute Language
- Check out the Tech Talk:

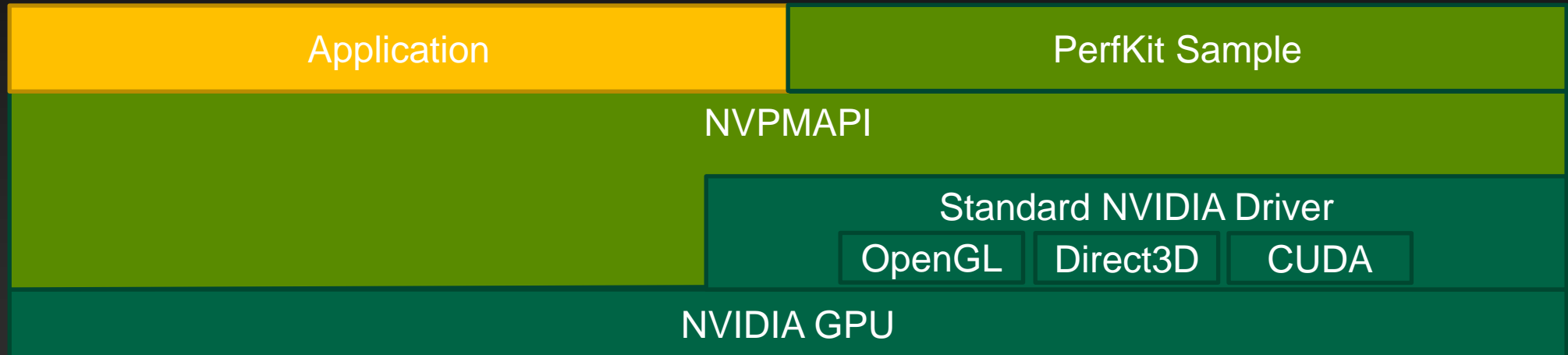
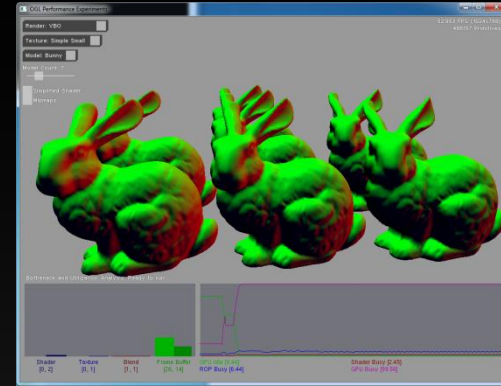
11:50 AM - 12:50 PM NVIDIA OpenGL in 2012
Mark Kilgard, Principal Software Engineer, NVIDIA

NVWMI

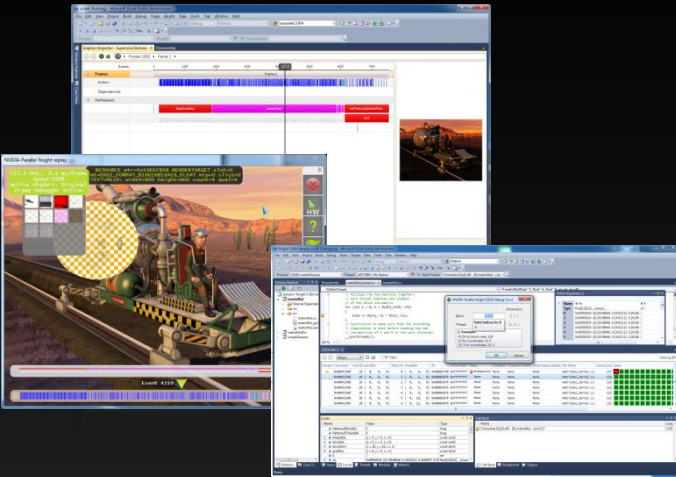
- **NVIDIA Windows Management Instrumentation infrastructure**
- **Exposes GPU information and configuration to network control**
- **Installs with every NVIDIA driver on Quadro GPUs**
 - Option during install

New NVIDIA® PerfKit

- GPU and Software Performance Counter API
 - Performance Monitoring
 - Automated bottleneck analysis
 - Graphics and Compute
- Support Kepler and Fermi Architecture
- Available on Windows (Linux release very soon), no special driver needed

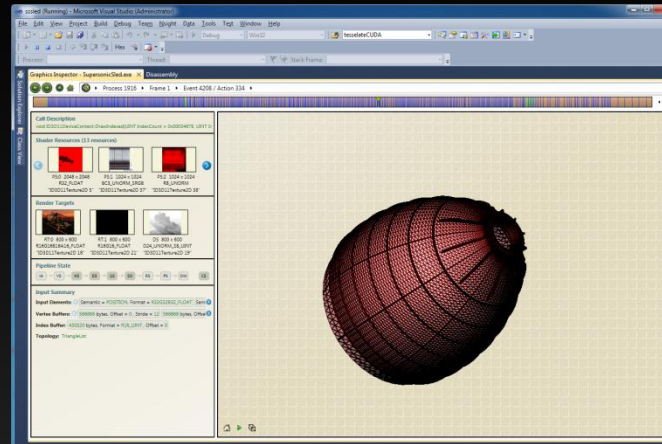


http://www.nvidia.com/object/nvperfkit_home.html



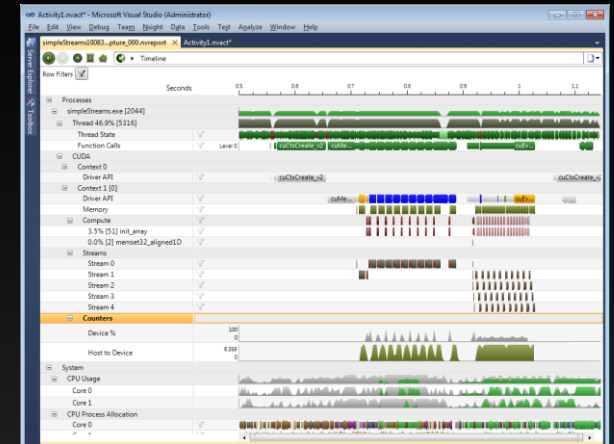
GPU Debugger

- GPU native Compute and Graphics debugging
- GPU breakpoints including complex conditionals
- GPU memory views and exception reporting
- Dynamic Shader Editing



Graphics Inspector

- Real-time inspection of 3D API calls
- Investigate GPU pipeline states
- See contributing fragments with Pixel History
- Profile frames to find GPU bottlenecks



System Analysis

- View CPU & GPU events on a single timeline
- Examine workload dependencies, memory transfers
- CPU/OS, Compute and Graphics APIs Trace
- Capture call stack and jump to source

Nsight™ Visual Studio Edition

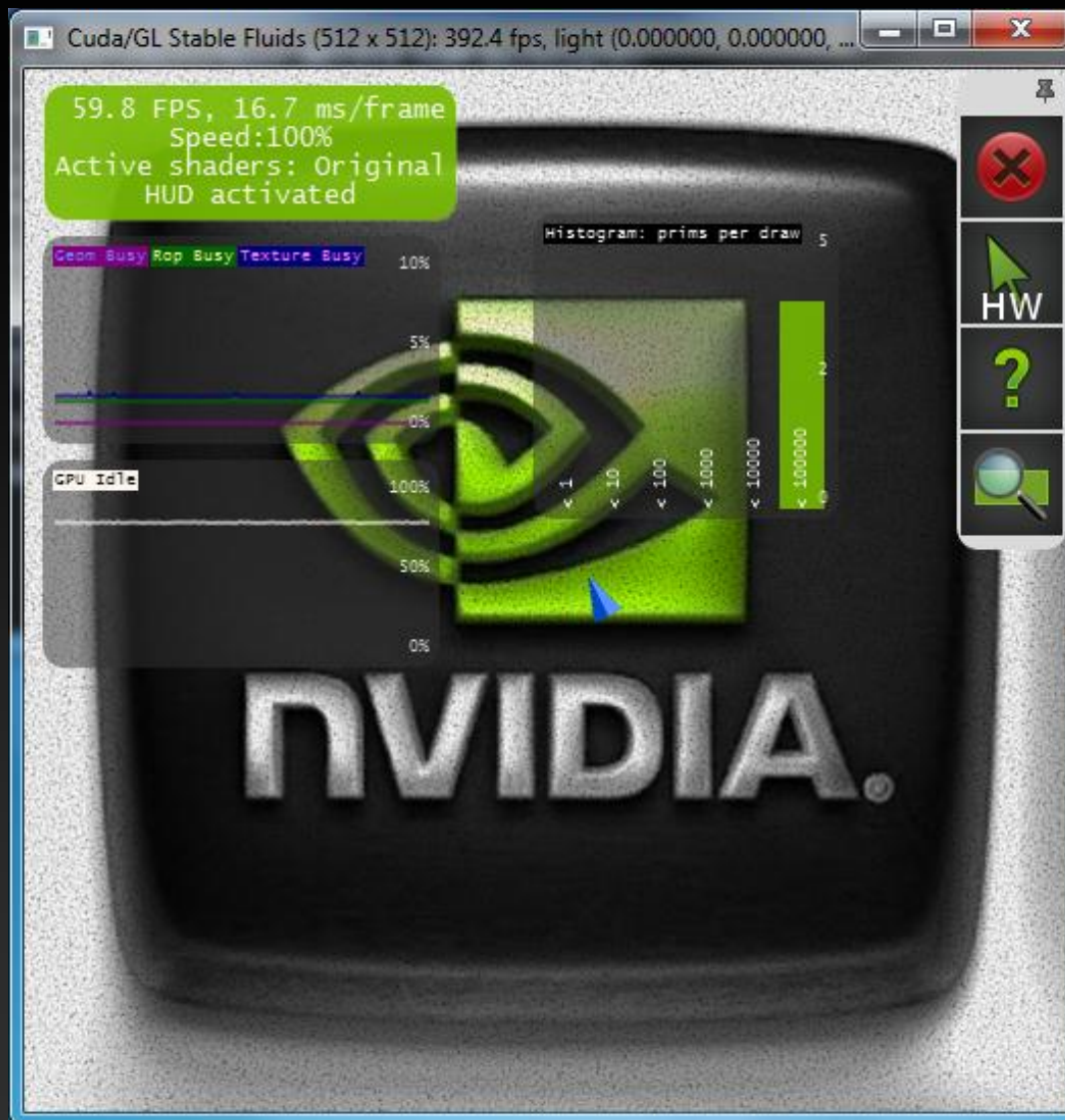
Release Codename “Hulk”

Perfect Maximus development companion

- Full support for CUDA® 5.0 and Kepler II (GK110)
- Full support for modern OpenGL (4.2 Core only for now)
- Debug CUDA and Graphics in the same session
- Single GPU Debugging for Graphics and Compute
- Advanced source-code-correlated CUDA® profiling



Nsight OpenGL HUD



GLSL Shader Debugging

The screenshot shows the Visual Studio IDE with the following components:

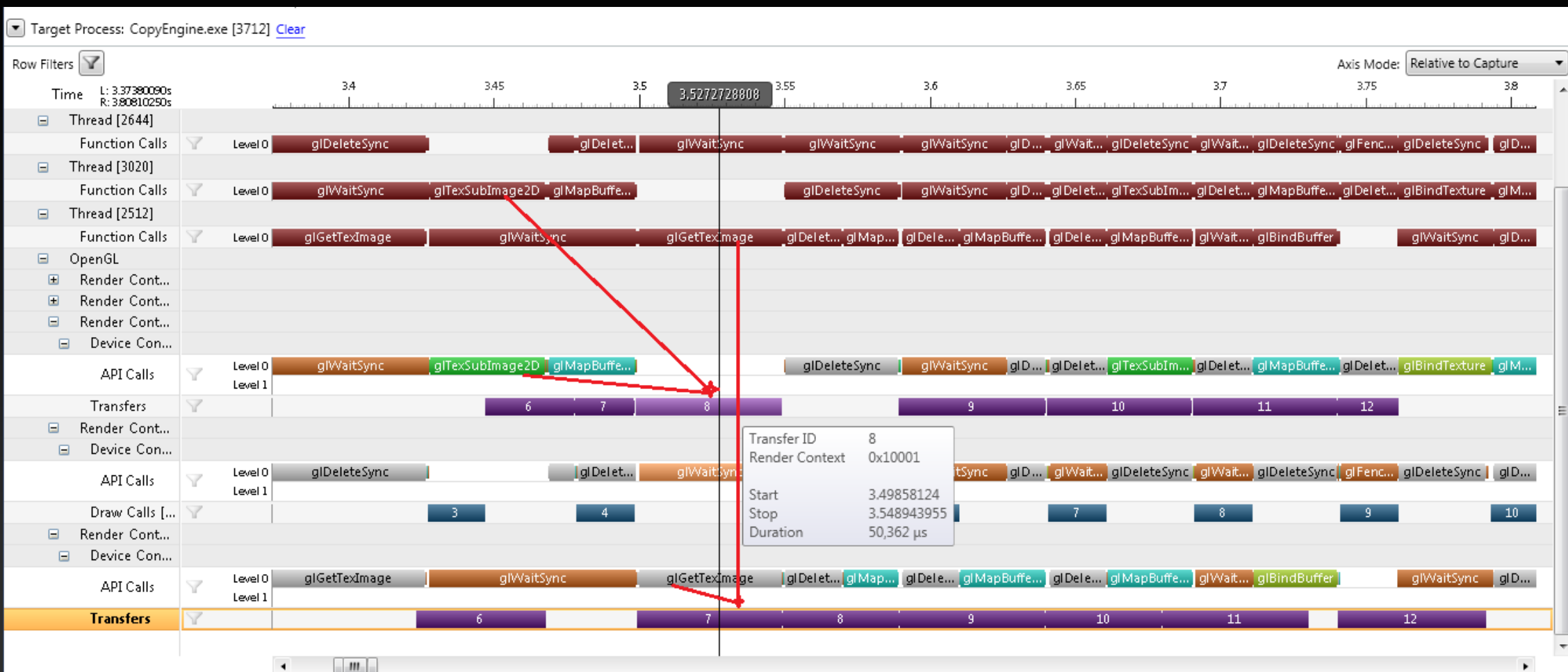
- Code Editor:** Displays GLSL code for a shader. The code includes uniform declarations, texture sampling, and vector calculations.
- Shaders Window:** Lists loaded shaders with their names, types, and statuses.
- Autos Window:** Shows the current state of variables in the current scope.
- Locals Window:** Shows the state of local variables, including texture coordinates and vectors.

```
6 uniform vec3 LightPos;
7
8
9 in block
10 {
11     vec2 TexCoord;
12 } In;
13
14 layout(location = 0, index = 0) out vec4 Color;
15
16
17 void main()
18 {
19     vec2 screenSpaceCoord = vec2(gl_FragCoord.x / 512.0f, 1.0 - (gl_FragCoord.y / 512.0f));
20     vec4 fragPos = vec4(gl_FragCoord.x / 512.0f * 2.0 - 1.0, (gl_FragCoord.y / 512.0f) * 2.0 - 1.0, 0.0f, 1.0f);
21     vec4 decalMap = texture(Diffuse, In.TexCoord);
22     vec4 normalMap = texture(NormalMap, screenSpaceCoord);
23     vec4 decalMapHere = texture(Diffuse, screenSpaceCoord);
24     vec3 lightDir = vec3(LightPos.x - fragPos.x, LightPos.y - fragPos.y, LightPos.z - fragPos.z);
25     vec3 viewDir = normalize(vec3(- fragPos.x, - fragPos.y, 10.0f));
26     vec3 v = normalize(viewDir);
27
28     vec4 heightVec = texture(HightMap, In.TexCoord);
29     float height = length(heightVec.xyz);
30
31     // Convert normal to -1,1
32     vec3 normal = normalize(normalMap.xyz * 2.0 - 1.0);
33 }
```

Name	Value	Type
height	1.3367199	float
heightVec	{x = 0.61960787, y = 0.61960787, z = 0.61960787, w = 1}	vec4
heightVec.xyz	{x = 0.61960787, y = 0.61960787, z = 0.61960787}	vec3

Name	Value	Type
screenSpaceCoord	{x = 0.032226563, y = 0.010742188}	vec2
fragPos	{x = -0.93554688, y = 0.97851563, z = 0, w = 1}	vec4
decalMap	{x = 1, y = 1, z = 1, w = 1}	vec4
normalMap	{x = 0.49938202, y = 0.48233768, z = 0.99607843, w = 1}	vec4
decalMapHere	{x = 1, y = 1, z = 1, w = 1}	vec4
lightDir	{x = 0.93554688, y = -0.97851563, z = 1}	vec3
viewDir	{x = 0.092708983, y = -0.096967012, z = 0.9909603}	vec3
v	{x = 0.092708983, y = -0.096967012, z = 0.9909603}	vec3
heightVec	{x = 0.61960787, y = 0.61960787, z = 0.61960787, w = 1}	vec4
height	1.3367199	float
normal	{x = 0.071822532, y = -0.34660852, z = 0.93525618}	vec3
newTexCoord	{x = 0, y = 0}	vec2
h	{x = -0.31798625, y = -0.3784112, z = 0.86930412}	vec3
nDotL	0.7489093	float
nDotH	0.92134404	float
power	0	float
ambient	{x = 0.2, y = 0.2, z = 0.2, w = 1}	vec4

Trace OpenGL async memory transfers



Hulk Source Code Correlation (WIP)

Tessellation120427...apture_000.nvreport

CUDA Source View

tessellateNURBSPatches_Kernel Grid Dim: {100, 1, 1} Block Dim: {512, 1, 1} Duration: 1406.4 μs

File: NURBSTessellation4.cu View: Source and SASS High to Low: Low to High:

Line	Source	Instructions Executed	Thread Instructions Executed	Thread Instructions Executed (False)	Thread Instructions Predicated Off
88	//return the index to the knot span that u is co...				
89	__device__ unsigned int findSpan(float u, float*...				
90	{				
91	for(unsigned int i = 0; i < nKnots - 1; i++)	22800	689700	96800	14.0
92	{				
93	if(u >= ku[i] && u < ku[i+1])	29200	883300	169400	19.2
94	return i;				
95	}				
96	return 0;				
97	}				
98					
99	__global__ void tessellateNURBSPatches_Kernel(NUR...				
100	{				
101	NURBSPatch *patch = &patches[blockIdx.X];	1600	51200	0	0.0

Line	Source	Instructions Executed	Thread Instructions Executed	Thread Instructions Executed (False)	Thread Instructions Predicated Off
931	/*1D10*/ @P0 BRA 0x1ac8; # Target=0x00001ac8	1600	51200	51200	100.0
932	/*1D18*/ EXIT;	1600	51200	0	0.0
933	/*1D20*/ IADD R5, R5, 0x1;	1200	36300	0	0.0
934	/*1D28*/ MOV R9, RZ;	1200	36300	0	0.0
935	/*1D30*/ ISETP.GE.U32.AND P0, PT, R5, R4, PT;	1200	36300	0	0.0
936	/*1D38*/ @P0 BRK;	1200	36300	36300	100.0
937	/*1D40*/ BRA 0x11f8; # Target=0x000011f8	1200	36300	0	0.0
938	/*1D48*/ IADD R5, R5, 0x1;	1200	36300	0	0.0
939	/*1D50*/ MOV R3, RZ;	1200	36300	0	0.0
940	/*1D58*/ ISETP.GE.U32.AND P0, PT, R5, R4, PT;	1200	36300	0	0.0
941	/*1D60*/ @P0 BRK;	1200	36300	36300	100.0
942	/*1D68*/ BRA 0x1160; # Target=0x00001160	1200	36300	0	0.0
943	/*1D70*/ IADD R1, R1, -0x28;	34000	202800	0	0.0
944	/*1D78*/ S2R R0, SR_LMemHiOff;	34000	202800	0	0.0

Drag a column header and drop it here to group by that column

File	Line #	Instructions Executed	Thread Instructions Executed	Thread Instructions Executed (False)	Thread Instructions Predicated Off	Active Mask	Predicates
nurbstessellation4.cu	66	47200	178200	0	0.0		
nurbstessellation4.cu	66	47200	178200	0	0.0		
nurbstessellation4.cu	66	34000	202800	0	0.0		
nurbstessellation4.cu	66	34000	202800	0	0.0		
nurbstessellation4.cu	66	34000	202800	0	0.0		
nurbstessellation4.cu	66	34000	202800	0	0.0		
nurbstessellation4.cu	13	34000	202800	0	0.0		

tessellateNURBSPatches_Kernel [CUDA Launch]

- tessellateNURBSPatches_Kernel [CUDA Kernel]
 - Experiment Results
 - CUDA Occupancy
 - CUDA Code Correlation
 - CUDA Instruction Count

OpenGL Insight

- Full debugging for OpenGL & Compute on a single GPU, including GLSL shaders
 - Shader break points!

- Check out the Tech Talk:

10:40 AM - 11:40 AM Enabling the Next-Generation of Computational Graphics with Nsight Visual Studio Edition

Jeff Kiel, Manager, Graphics Related Developer Tools, NVIDIA

Summary



- **Kepler architecture offers some significantly improved architectural efficiency**
- **Many new features and capabilities for developers**
- **Graphics combined with compute becoming increasingly prevalent**
- **Powerful new developer tools to compliment Kepler architecture**

GTC 2013 | March 18-21 | San Jose, CA

The Smartest People. The Best Ideas. The Biggest Opportunities.

Opportunities for Participation:

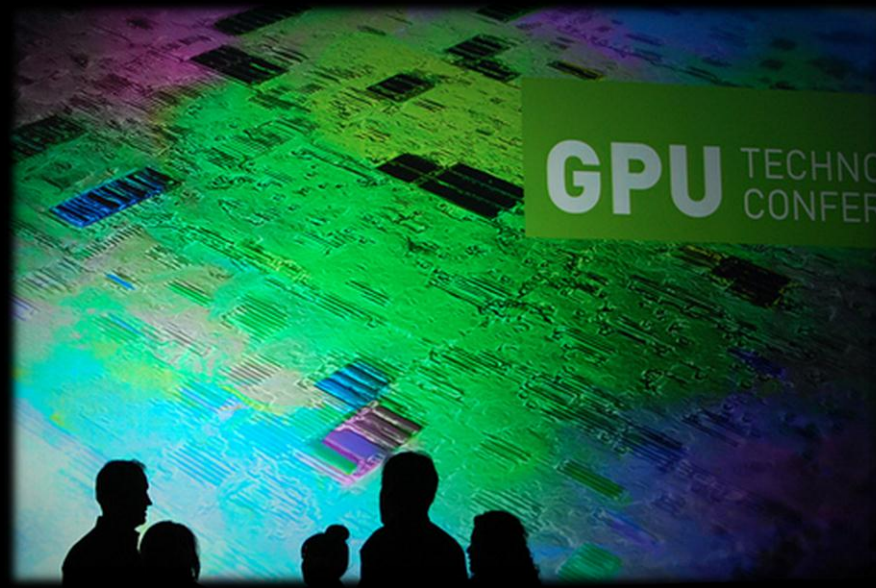
SPEAK - Showcase your work among the elite of graphics computing

- Call for Sessions: August 2012
- Call for Posters: October 2012

REGISTER - learn from the experts and network with your peers

- Use promo code **GM10SIGG** for a 10% discount

SPONSOR - Reach influential IT decision-makers



Learn more at www.gputechconf.com