Performance Tools Agenda

- Overview of GPU pipeline and Unified Shader
- NVIDIA PerfKit 5.0: Driver & GPU Performance Data
  - PerfHUD: The GPU Performance Accelerator
  - PerfSDK: Performance data integrated into your application
- ShaderPerf: Shader program performance
GPU Pipelined Architecture (Logical View)

- CPU
- Vertex Assembly
- Vertex Shader
- Geometry Shader
- Rasterizer
- Pixel Shader
- Blending

Texture
Framebuffer
GPU Pipelined Architecture (Logical View)
**Common Problems**

- New, more complex GPU hardware
  - GPU is a black box
  - Unified shaders changes everything
- Increasing engine and scene complexity
- Artists don’t always understand how rendering engines work
- CPU tuning only gets you so far
- Turn around time for debugging and tuning shaders is too long
- Hard to debug setup issues
PerfHUD solves issues for top developers!

In a recent survey, over 100 PerfHUD 4 users reported an average speedup of 35%, and as much as 400% in some cases!
PerfHUD: Direct3D debugging and tuning

- One click bottleneck determination
- Graphs and debugging tools overlaid on your application
- 4 screens for targeted analysis
  - Performance Dashboard
  - Debug Console
  - Frame Debugger
  - Frame Profiler
- Drag and drop application on PerfHUD icon
New! PerfHUD 5.0!

- Interactive model
  - Shader Edit and Continue
  - Render state Modification
  - Configurable Graphs
  - Many more features and usability improvements

- New technologies
  - Windows Vista & DirectX 10
  - GeForce 8800 and Unified Shaders
Demo: PerfHUD
Demo: Performance Dashboard
Demo: Performance Dashboard
Demo: Frame Debugger
Demo: Advanced Frame Debug
Frame Profiler

- PerfHUD uses PerfKit and SimExp
- Sample ~35 performance counters
- Multiple passes on the scene since they can’t be at the same time
- Need to render THE SAME FRAME until all the counters are read
  - Only possible if application uses time-based animation
  - Intercept: QueryPerformanceCounter(), timeGetTime()
  - NO RDTSC!!
Demo: Frame Profiler
Demo: Advanced Frame Profiler
Counter Types

- SW/Driver Counters: PerfAPI, PDH
- Raw GPU Counters: PerfAPI, PDH
- Simplified Experiments: PerfAPI
- Instrumented GPUs

- GeForce 8800 Series
- GeForce 7950/7900 GTX & GT
- Quadro FX 5500 & 4500

- GeForce 7800 GTX
- GeForce 6800 Ultra & GT
- GeForce 6600

© NVIDIA Corporation 2007
Direct3D/OpenGL Driver Counters

**General**
- FPS
- ms per frame

**Driver**
- Driver frame time (total time spent in driver)
- Driver sleep time (waiting for GPU)
- % of the frame time driver is waiting

**Counts**
- Batches, vertices, primitives
- (Direct3D) Triangles and instanced triangles
- (Direct3D) Locked render targets

**Memory**
- AGP memory used
- Video memory used and total
GPU Counters

- **gpu_idle**
- **vertex_attribute_count**
- ** shader_busy**
  - vertex, geometry, pixel ratios
- **culled_primitive_count**
  - primitive_count
  - triangle_count
  - vertex_count
- **shaded_pixel_count**
- **rop_busy**

**Texture Unit (Filtering)**

**Frame Buffer (RAM Memory)**

**Vertex Assembly**

**Vertex Shader**

**Geometry Shader**

**Raster / ZCull**

**Pixel Shader**

**Raster Operations**

© NVIDIA Corporation 2007
How do I use PerfKit counters?

- **PerfAPI**: Easy integration of PerfKit
  - Real time performance monitoring using GPU and driver counters, round robin sampling
  - Simplified Experiments for single frame analysis

- **PDH**: Performance Data Helper for Windows
  - Driver data, GPU counters, and OS information
  - Exposed via Perfmon
  - Good for rapid prototyping

- Sample code and helper classes provided in PerfSDK
PerfAPI: Real Time

// Somewhere in setup
NVPMAddCounterByName("vertex_shader_busy");
NVPMAddCounterByName("pixel_shader_busy");
NVPMAddCounterByName("shader_waits_for_texture");
NVPMAddCounterByName("gpu_idle");

// In your rendering loop, sample using names
NVPMSample(NULL, &nNumSamples);
NVPMGetCounterValueByName("vertex_shader_busy", 0, &nVSEvents, &nVSCycles);
NVPMGetCounterValueByName("pixel_shader_busy", 0, &nPSEvents, &nPSCycles);
NVPMGetCounterValueByName("shader_waits_for_texture", 0, &nTexEvents, &nTexCycles);
NVPMGetCounterValueByName("gpu_idle", 0, &nIdleEvents, &nIdleCycles);
PerfAPI: SimExp
Associated Tools: NVIDIA Plug-In for Microsoft PIX for Windows
Graphic Remedy’s gDEBugger

- OpenGL Debugging and Performance Tool
  - PerfKit and GLExpert integrated
  - Version 3.0 to support Vista
  - Linux version in Beta, release soon
  - Free academic licenses available from Graphic Remedy and the OpenGL ARB.
  - One year license for full featured version, including all software updates

Details: [http://academic.gremedy.com](http://academic.gremedy.com)
Project Status

- PerfKit 4.2 for Windows 32bit available: developer.nvidia.com
- PerfKit 5.0 (Q2 2007)
  - PerfHUD 5.0
  - ForceWare Release 100 Driver
  - GeForce 8800 support
  - Windows 32 and 64 bit
  - Linux 32 and 64 bit
- PerfGraph: www.sourceforge.org\perfgraph
- Instrumented GPUs
  - GeForce 8800 Series
  - GeForce 7950/7900 GTX & GT
  - Quadro FX 5500 & 4500
  - GeForce 7800 GTX
  - GeForce 6800 Ultra & GT
  - GeForce 6600

Feedback and Support: NVPerfKit@nvidia.com
v2f BumpReflectVS(a2v IN,
    uniform float4x4 WorldViewProj,
    uniform float4x4 World,
    uniform float4x4 ViewIT)
{
    v2f OUT;// Position in screen space.OUT.Position = mul(IN.Position, WorldViewProj);// pass texture coordinates for fetching the normal mapOUT.TexCoord = IN.TexCoord;OUT.TexCoord.w = 1.0;// compute the 4x4 transform from tangent space to object spacefloat3x3 TangentToObjSpace;// first rows are the tangent and binormal scaled by the bump scaleTangentToObjSpace[0] = IN.Tangent.y, IN.Binormal.y, IN.Normal.y);TangentToObjSpace[2] = float3(IN.Tangent.z, IN.Binormal.z, IN.Normal.z);OUT.TexCoord1.x = dot(World[0].xyz, TangentToObjSpace[0]);OUT.TexCoord1.y = dot(World[1].xyz, TangentToObjSpace[0]);OUT.TexCoord1.z = dot(World[2].xyz, TangentToObjSpace[0]);OUT.TexCoord2.x = dot(World[0].xyz, TangentToObjSpace[1]);OUT.TexCoord2.y = dot(World[1].xyz, TangentToObjSpace[1]);OUT.TexCoord2.z = dot(World[2].xyz, TangentToObjSpace[1]);OUT.TexCoord3.x = dot(World[0].xyz, TangentToObjSpace[2]);OUT.TexCoord3.y = dot(World[1].xyz, TangentToObjSpace[2]);OUT.TexCoord3.z = dot(World[2].xyz, TangentToObjSpace[2]);OUT.TexCoord1.w = eyeVector.x;OUT.TexCoord2.w = eyeVector.y;OUT.TexCoord3.w = eyeVector.z;return OUT;
}

///////////////// pixel shader //////////////////
float4 BumpReflectPS(v2f IN,
    uniform sampler2D NormalMap,
    uniform samplerCUBE EnvironmentMap,
    uniform float BumpScale) : COLOR
{
    // fetch the bump normal from the normal mapfloat3 normal = tex2D(NormalMap, IN.TexCoord.xy).xyz * 2.0 - 1.0;normal = normalize(normal);float3 worldNorm;worldNorm.x = dot(IN.TexCoord1.xyz, normal);worldNorm.y = dot(IN.TexCoord2.xyz, normal);worldNorm.z = dot(IN.TexCoord3.xyz, normal);worldNorm.w = dot(IN.TexCoord1.w, normal);worldNorm = normalize(worldNorm);worldNorm = normalize(worldNorm * BumpScale);float3 eyevec = float3(IN.TexCoord1.w, IN.TexCoord2.w, IN.TexCoord3.w);float3 lookup = reflect(eyevec, worldNorm);return texCUBE(EnvironmentMap, lookup);
}

ShaderPerf 2.0

Inputs:
• GLSL, Cg, HLSL
• PS1.x, PS2.x, PS3.x
• VS1.x, VS2.x, VS3.x
• !!FP1.0
• !!ARBfp1.0

GPU Arch:
• GeForce 7X00
• GeForce 6X00
• GeForce FX series
• Quadro FX series

Outputs:
• Resulting assembly code
• # of cycles
• # of temporary registers
• Pixel/vertex throughput
• Test all fp16 and all fp32
ShaderPerf: In your pipeline

- Test current performance
  - Compare with shader cycle budgets
  - Test optimization opportunities
  - Not just Tex/ALU balance: cycles & throughput

- Automated regression analysis

- Integrated in FX Composer 2.0
  - Artists/TDs code expensive shaders
  - Achieve optimum performance
ShaderPerf 2.0 Alpha

- Supports Direct3D/HLSL
- GeForce 7XXX, 6XXX, and FX GPUs
- ForceWare Release 100 Unified Compiler
- Improved vertex performance simulation and throughput calculation
- Multiple drivers from one ShaderPerf
- Smaller footprint
- New programmatic interface
Shader Perf 2.0: Beta

- Full support for Cg and GLSL, vertex and fragment programs
- Support for GeForce 8XXX series GPUs
- Geometry shaders and geometry throughput
Questions?

Developer tools DVDs available at our booth
- PerfKit 2.2
- PerfHUD 4 Overview Video
- PerfHUD 4 Quick Reference Card
- ShaderPerf 2.0 Alpha
- User Guides

Online:

Feedback and Support:
- NVPerfKit@nvidia.com
- NVPerfHUD@nvidia.com
- NVShaderPerf@nvidia.com
- FXComposer@nvidia.com
Six All-New NVIDIA Developer Tools!
Check it out at booth #5134!

- SDK 10
- PerfKit 5
- FX Composer 2
- GPU-Accelerated Texture Tools
- ShaderPerf 2
- Shader Library