Generic CUDA debugger goals

- Support all standard debuggers across all OS
  - Linux – GDB, TotalView and DDD
  - Windows – Visual studio
  - Mac - XCode
- Support CUDA runtime APIs
- Support CUDA driver APIs
- Unified debug environment
  - Provide simultaneous access to CUDA and host variables
    - Present CUDA threads same as host threads
    - Present CUDA memory same as host memory
- Support single and multi GPU debugging
- Attach to a running process – local or remote.
IPC – Driver APIs to Debug APIs

- Driver API callbacks retrieves the fatbin/SASS line information
- Debug APIs translates the SASS information to actual GPU address
- Debug APIs tracks all memory allocations
- Driver and Debug APIs report and manage kernel launch, execution and termination.
Current Status

- CUDA-GDB 3.0 Beta shipping to customers supporting:
  - New CUDA Memory Checker
  - Support for all the OpenCL features
  - Early support for Fermi Architecture
  - etc…
Upcoming Roadmap

- Migrate to standard binary formats – DWARF and ELF
- Performance Improvement
- New debug APIs to enable external customers
- Support source and assembly level debugging
- Support for Apps with multiple host threads using CUDA
- Support for Apps using multi-GPUs
## Windows Development

### Development Environment

<table>
<thead>
<tr>
<th>Visual Studio</th>
<th>CPU Code</th>
<th>Tools</th>
<th>GPU Code</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>System</td>
<td>Project Wizards</td>
<td>Shader</td>
</tr>
<tr>
<td></td>
<td>Rendering</td>
<td>Editor</td>
<td>CUDA</td>
</tr>
<tr>
<td></td>
<td>AI</td>
<td>Debugger</td>
<td>OpenCL</td>
</tr>
<tr>
<td></td>
<td>Sound</td>
<td>Performance</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gameplay</td>
<td>Build System</td>
<td></td>
</tr>
</tbody>
</table>

- Shader Authoring
- Graphics Debugging
- Graphics Performance
- CUDA Performance
NVIDIA Nexus

Next-generation
development toolset

Homogeneous development
for CPU and GPU

Seamless debugging, profiling,
and visualization
The Nexus 1.0 Release

A full Visual Studio-integrated development environment

Supporting
CUDA C, DirectCompute, OpenCL, DirectX 11, DirectX 10, OpenGL

Requires Windows Vista or Windows 7, Visual Studio 2008 SP1
The Nexus Ecosystem

Development Environment

Visual Studio

- CPU Code
  - System
  - Rendering
  - AI
  - Sound
  - Gameplay

- Tools
  - Build System
  - Editor
  - Debugger
  - Performance
  - Gfx Analysis
  - CUDA Analysis
  - PhysX Analysis

- GPU Code
  - Shader
  - CUDA
  - OpenCL*

Shader Authoring
Code Authoring

Strong integration with the Visual Studio Shell.

Syntax highlighting

Code completion

Nexus toolbar, menu, and project wizards
Step through GPU code, and examine program values.

```
extern "C"
广阔 void
matrixMul( float* C, float* A, float* B, int wA, int wB)
{
    // Block index
    int bx = blockIdx.x;
    int by = blockIdx.y;

    // Thread index
    int tx = threadIdx.x;
    int ty = threadIdx.y;

    // Index of the first sub-matrix of A processed by the block
    int aBegin = wA * BLOCK_SIZE * by;

    // Index of the last sub-matrix of A processed by the block
    int aEnd = aBegin + wA - 1;

    // Step size used to iterate through the sub-matrices of A
    int aStep = BLOCK_SIZE;

    // Index of the first sub-matrix of B processed by the block
    int bBegin = BLOCK_SIZE * bx;

    // Step size used to iterate through the sub-matrices of B
    int bStep = BLOCK_SIZE * wB;

    for (int ia = aBegin; ia < aEnd; ia += aStep)
    {
        for (int ib = bBegin; ib < bBegin + wB; ib += bStep)
        {
            // Compute each element C[ia][ib]
            C[ia] += A[ia][ib] * B[ib];
        }
    }
}
```
Code Debugging

Power-up your breakpoints with parallel-aware conditionals.

<table>
<thead>
<tr>
<th>Name</th>
<th>Language</th>
<th>Hit Count</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageProcess.cu: Line 256</td>
<td>CUDA</td>
<td>break</td>
<td>BlockID(&gt; 1, &gt;= 1)</td>
</tr>
<tr>
<td>BufferProcessing BPs</td>
<td>CUDA</td>
<td>break</td>
<td>(no condition)</td>
</tr>
<tr>
<td>BufferProcess.cu: Line 39</td>
<td>CUDA</td>
<td>hits &gt; 3</td>
<td>(no condition)</td>
</tr>
<tr>
<td>BufferProcess.cu: Line 58</td>
<td>CUDA</td>
<td>hits &gt; 3</td>
<td>(no condition)</td>
</tr>
<tr>
<td>BufferProcess.cu: Line 109</td>
<td>CUDA</td>
<td>hits &gt; 3</td>
<td>(no condition)</td>
</tr>
<tr>
<td>BumpMapReflect: Line 18</td>
<td>HLSL10</td>
<td>break</td>
<td>Pixel(280, 580) NormalZ&gt;5</td>
</tr>
</tbody>
</table>
Code Debugging

Manage debugging across thousands of threads.
Graphics

Drill down from examining a frame...

Tabbed information viewer allows the user to explore the resource types over the entire frame. The view of each tab can be changed between a thumbnail view, and a tabular details view. (Like Windows Explorer)

The current draw call is previewed here, in Render Target or Draw Call mode. Render Target mode shows the current render target for the draw call with the draw call wireframe highlighted. Draw Call Geom mode shows the draw call geometry only, and can be rotated.
Graphics

...to a draw call...
Graphics

...to specific resources and GPU state.
Graphics

See GPU usage on a per-draw call basis.
Analysis

View performance from higher level using correlated CPU, GPU, thread, and OS data.
Analysis

Get detailed performance information for every kernel.
Nexus everywhere

Debug, visualize, and optimize your code remotely.

Visual Studio
Nexus
Host
Network
Nexus Monitor
Target Application
CUDA
DirectX
Target