Agenda

- Introduction to Parallel Nsight
- CUDA C/C++ Source Debugging
- Analysis/System Trace
- Q&A
What is Parallel Nsight?

- Development environment for heterogeneous platforms (CPU and GPU)
- Fully integrated into Microsoft Visual Studio 2008 and 2010
- Dramatic productivity improvement in common development tasks
Parallel Nsight for Compute

**Parallel Compute Debugger**
- Examine compute kernels directly on GPU hardware
- Debug CUDA C/C++ and DirectCompute applications
- Visualize thousands of threads executing in parallel using Visual Studio
- Use conditional breakpoints to correct errors in massively parallel code

**System Analyzer (Pro only)**
- Capture and visualize CPU and GPU level events on a single correlated timeline
- Inspect workload dependencies using the Timeline View
- Profile CUDA kernels using GPU performance counters
CUDA C/C++ Debugging

- Compile your code with Debug flag
- Use the familiar Visual Studio interface to debug your GPU code
- Dramatic productivity improvements
  - Explore memory during a live session vs. coding specific transfers
  - Immediately view live variables vs. printf/recompile loop
  - Set data breakpoints on memory area vs. trial and error
  - And much more!
Setting Breakpoints

```c
// 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 * ty;

// 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 * tx;

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

// C_sub is used to store the element of the block sub-matrix
// that is computed by the thread
float C_sub = 0;

// Loop over all the sub-matrices of A and B
// required to compute the block sub-matrix
for (int a = aBegin, b = bBegin; a <= aEnd; a += aStep, b += bStep) {
    // Declaration of the shared memory array A used to
    // store the sub-matrix of A
    // Declaration of the shared memory array B used to
    // store the sub-matrix of B
}
```
### Viewing Variable Values

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>a + wA * ty + tx</td>
<td>16</td>
<td>int</td>
</tr>
<tr>
<td>C[a + wA * ty + tx]</td>
<td>13.683188</td>
<td>_device__float&amp;</td>
</tr>
<tr>
<td>C[a + wA * ty + tx + 10]</td>
<td>13.49013</td>
<td>_device__float&amp;</td>
</tr>
<tr>
<td>C[16] + 34</td>
<td>47.683189</td>
<td>float</td>
</tr>
</tbody>
</table>
Viewing GPU Memory

Matrix multiplication kernel code:

```c
// Write the block sub-matrix to device memory;
// each thread writes one element
int c = wB * BLOCK_SIZE * by + BLOCK_SIZE * bx;
C[c + wB * ty + tx] = Csub;
```

```c
#else /* MATRIXMUL_KERNEL_H */
```

```c
#endif
```
Switching Between Threads

// Index of the last sub-matrix
int aEnd = aBegin + W - 1;

// Step size used to iterate through A
int aStep = BLOCK_SIZE;

// Index of the first sub-matrix
int bBegin = BLOCK_SIZE * bx;

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

// Csub is used to store the element of the block sub-matrix that is computed by the thread
float Csub = 0;

Dimensions

Block: 3, 8, 5, 1
Thread: Valid Indices for Y (Filtered) 0-4

Examples

#129 for block index 129
10 for coordinates 10.0
10, 5 for coordinates 10, 5
Conditional Breakpoints

// Declaration of the shared memory array As
// store the sub-matrix of A
__shared__ float As[BLOCK_SIZE][BLOCK_SIZE];

// Declaration of the shared memory array Bs
// store the sub-matrix of B
__shared__ float Bs[BLOCK_SIZE][BLOCK_SIZE];

// Load the matrices from device memory
// to shared memory; each thread loads
// one element of each matrix
AS(ty, tx) = A[a + wA * ty + tx];
BS(ty, tx) = B[b + wB * ty + tx];

// Synchronize to make sure the matrices are...
Data Breakpoints

```
// computation is done before loading two new
// sub-matrices of A and B in the next iteration
__syncthreads();

// Write the block sub-matrix to device memory;
// each thread writes one element
int c = wB * BLOCK_SIZE * by + BLOCK_SIZE * bx;
C[c + wB * ty + tx] = Csub;
```
Trace

- Powerful performance tool
- Correlated timeline between CPU and GPU
- Bottleneck identification
  - Find CPU vs. GPU boundedness
  - Find memory transfer vs. kernel computation bounded
  - Get macro-level information on which CUDA kernels use the most time
Trace across the CPU and GPU
Trace – Overlap Memory Transactions
Tesla Compute Cluster Support (TCC)

- TCC is a special driver mode for Windows 7, Vista, and HPC Server 2008.
  - Included in our most recent R260 driver release.

- Benefits
  - Lower overhead to kernel launches = higher performance
  - Running CUDA on a MS Remote desktop

- Parallel Nsight 1.5 now supports debugging on GPUs using a TCC driver.
4 Flexible GPU Development Configurations

**Desktop**
- Single machine, Single NVIDIA GPU
  - Analyzer, Graphics Inspector

**Networked**
- Two machines connected over the network
  - Analyzer, Graphics Inspector, Compute Debugger, Graphics Debugger

**Workstation SLI**
- SLI Multi OS workstation with two Quadro GPUs
  - Analyzer, Graphics Inspector, Compute Debugger, Graphics Debugger

**NEW**
- Single machine, Dual NVIDIA GPUs
  - Analyzer, Graphics Inspector, Compute Debugger
<table>
<thead>
<tr>
<th>Feature</th>
<th>Standard (no cost)</th>
<th>Professional ($349) available for purchase in December</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute Debugger</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DirectX 10 &amp; 11 Debugger &amp; Graphics inspector</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>GeForce Support: 9 series or higher</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Tesla Support: C1050/S1070 or higher</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Quadro Support: G9x or higher</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Windows 7, Vista and HPC Server 2008</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Visual Studio 2008 SP1 and Visual Studio 2010</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Compute Analyzer</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>OpenGL and OpenCL Analyzer</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>DirectX 10 &amp; 11 Analyzer</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Tesla Compute Cluster (TCC) Debugging</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

http://www.nvidia.com/GetParallelNsight
Parallel Nsight Resources

- Parallel Nsight GPU Computing Forum
- The Parallel Nsight User Guide
  - Installed with the Host installer
  - Available on the Web
- Links to these from: [http://developer.nvidia.com/ParallelNsight](http://developer.nvidia.com/ParallelNsight)