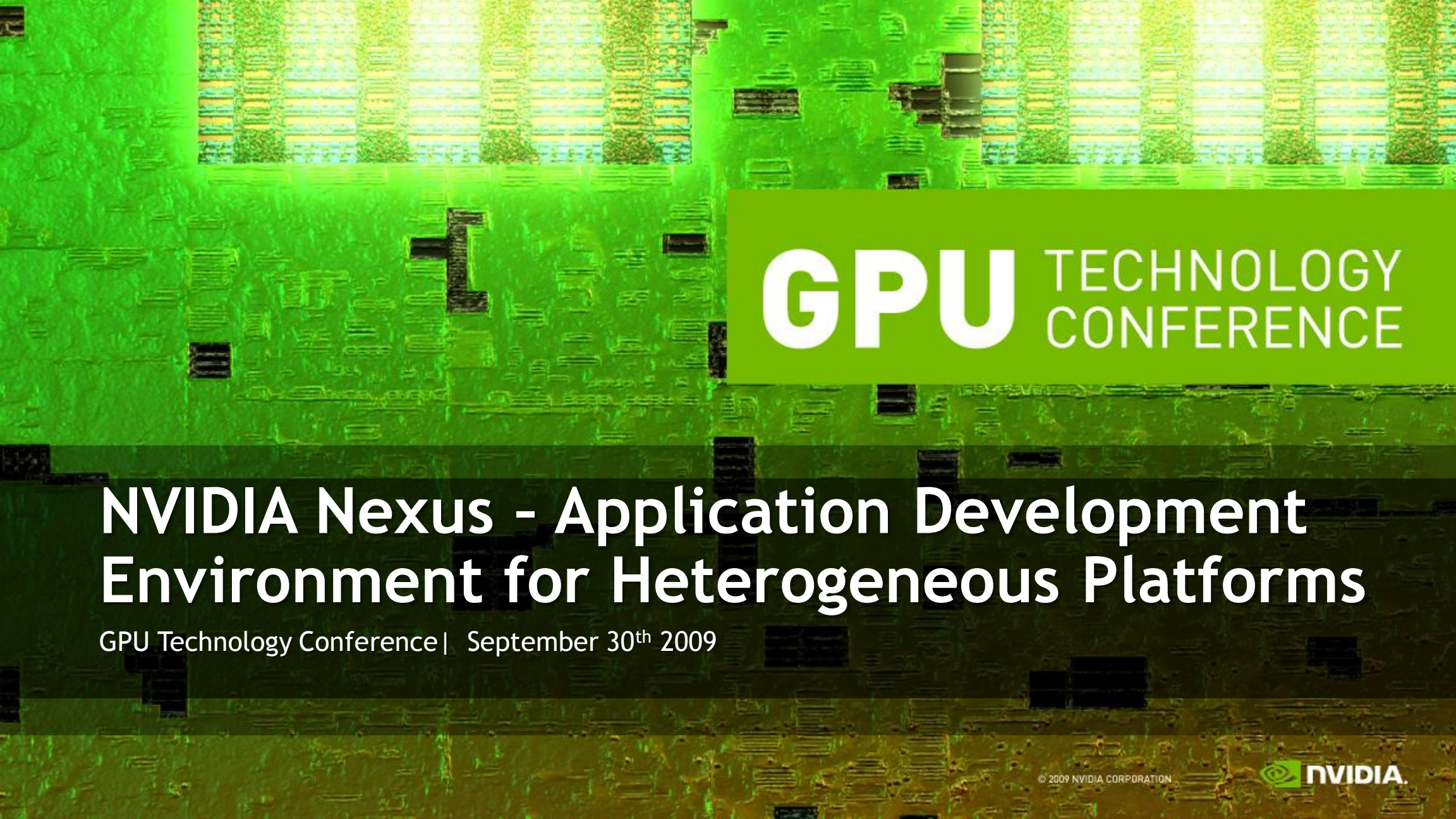


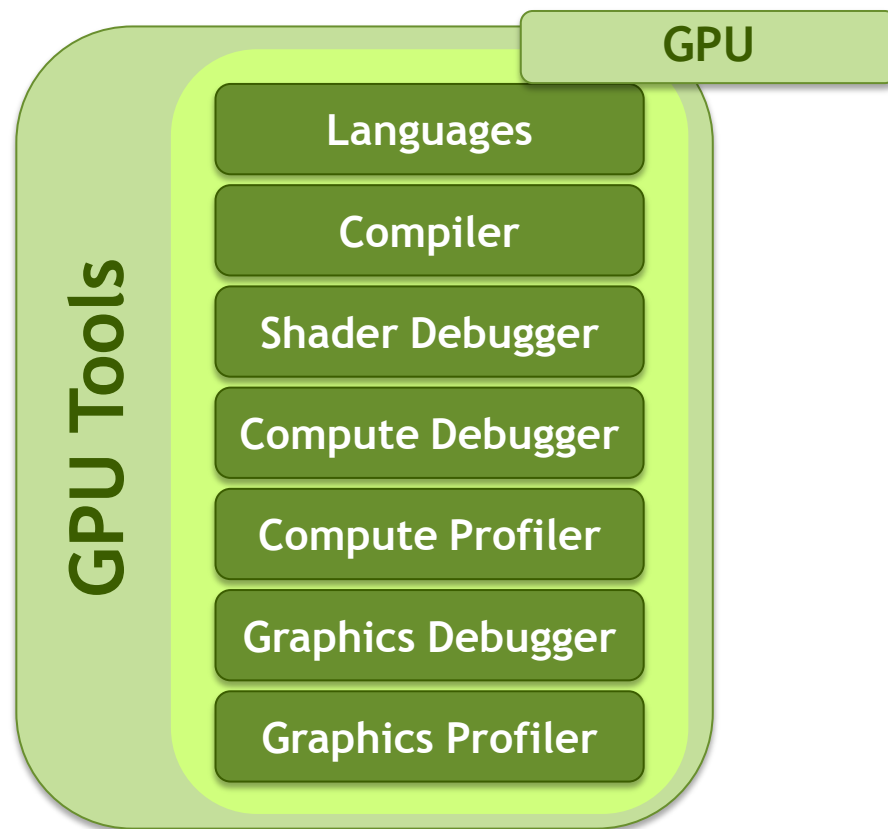
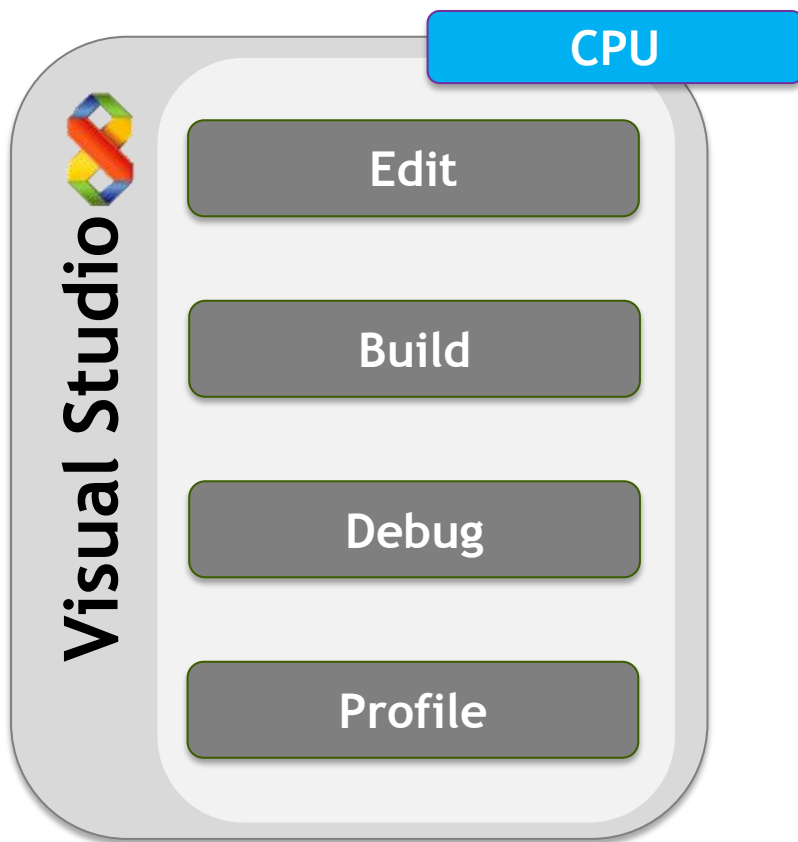


GPU TECHNOLOGY CONFERENCE

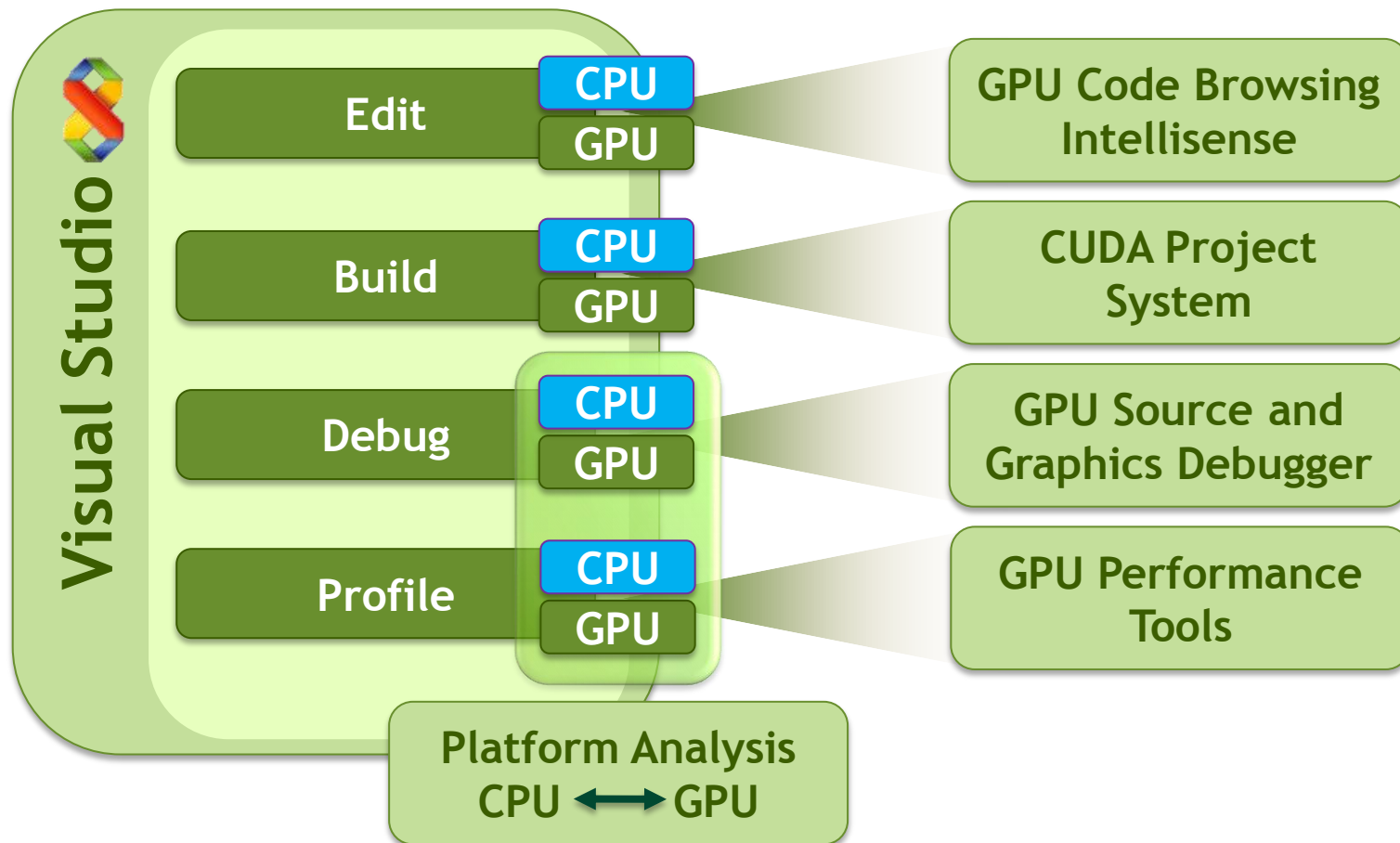
NVIDIA Nexus - Application Development Environment for Heterogeneous Platforms

GPU Technology Conference | September 30th 2009

Windows Development Environment



... what developers really want



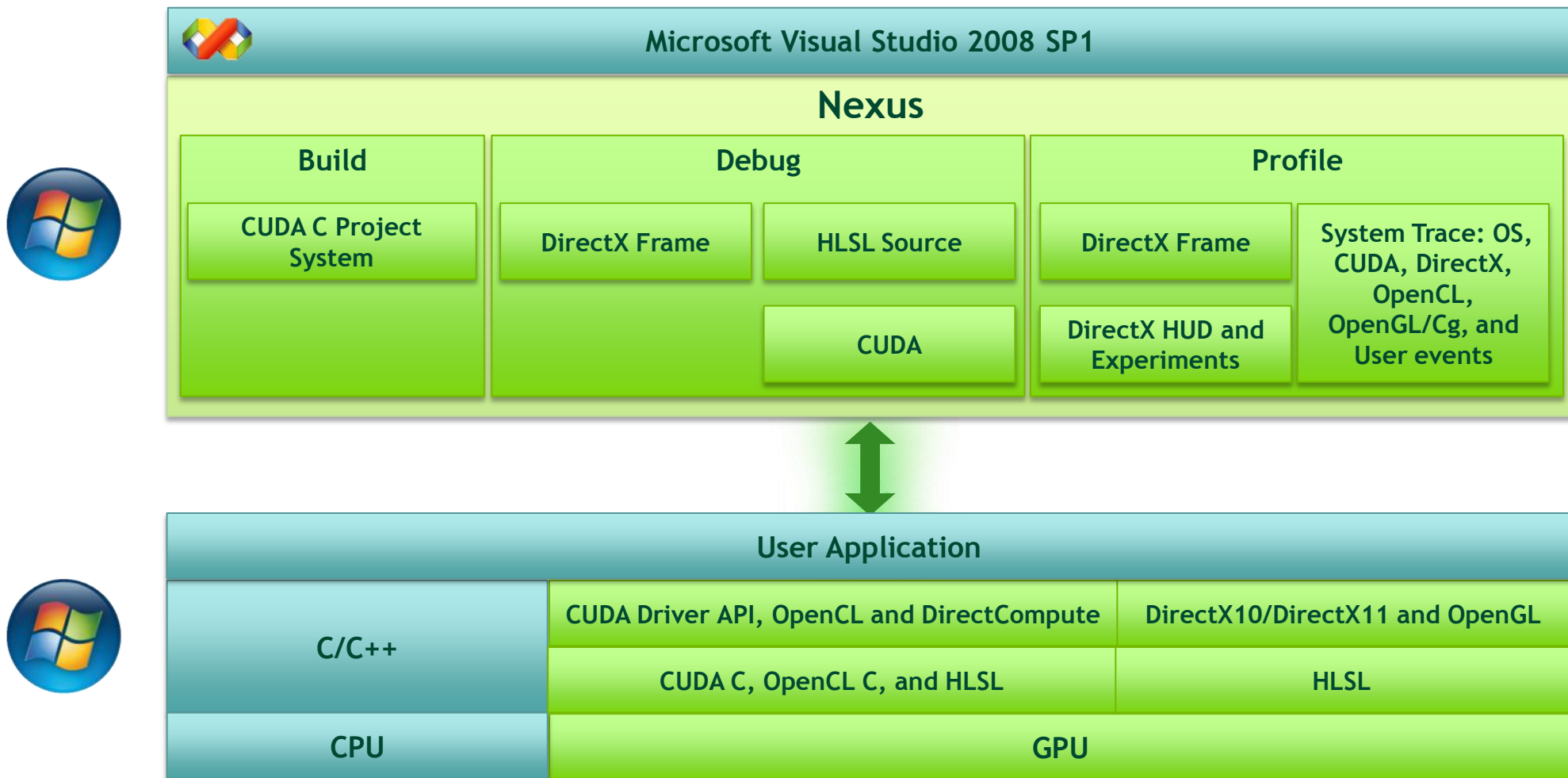
NVIDIA Nexus

Enables Seamless Co-Processing Development

- Full-featured debugging on GPU
- Platform-wide activity analysis
- Visual Studio 2008 integration



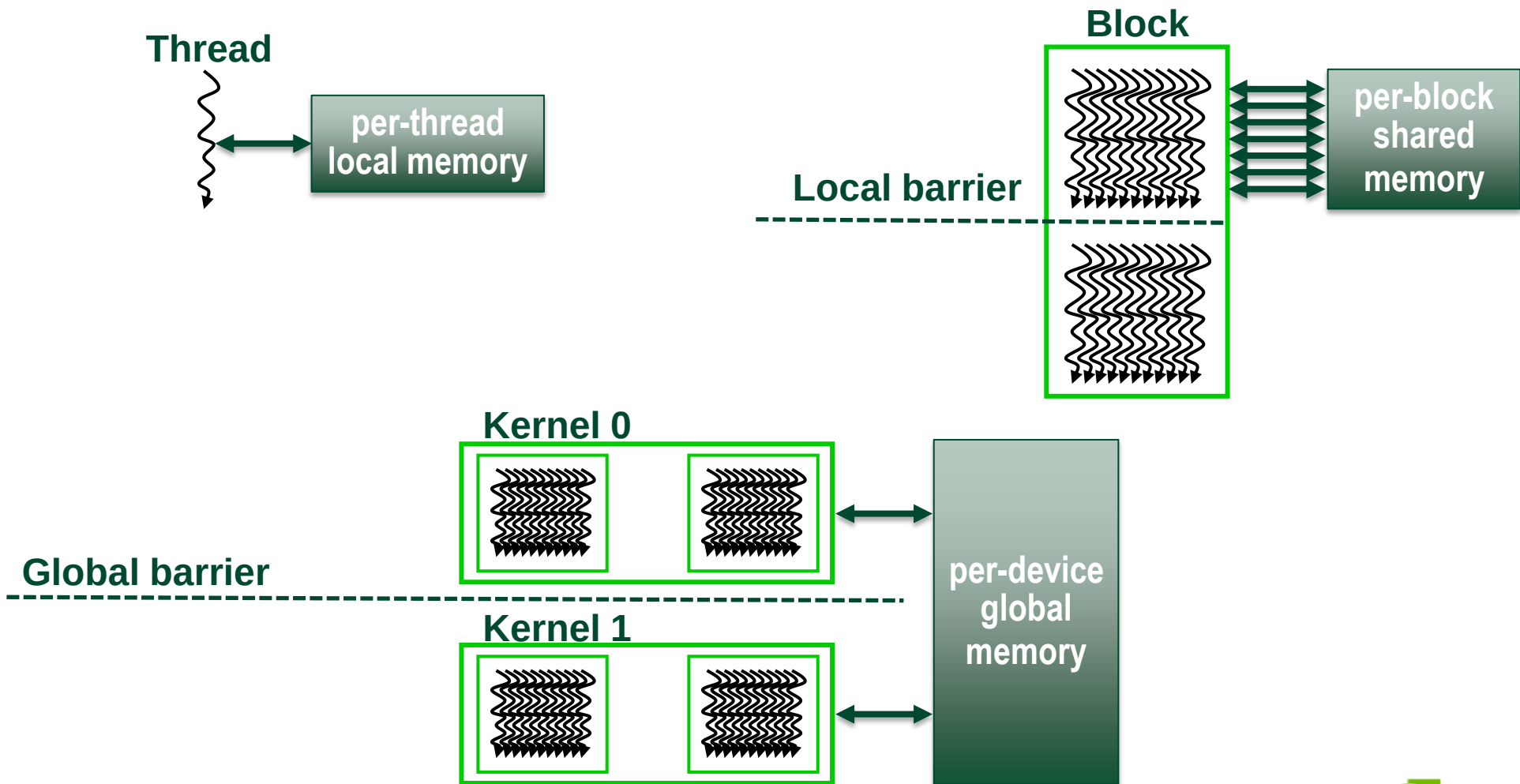
Nexus Overview



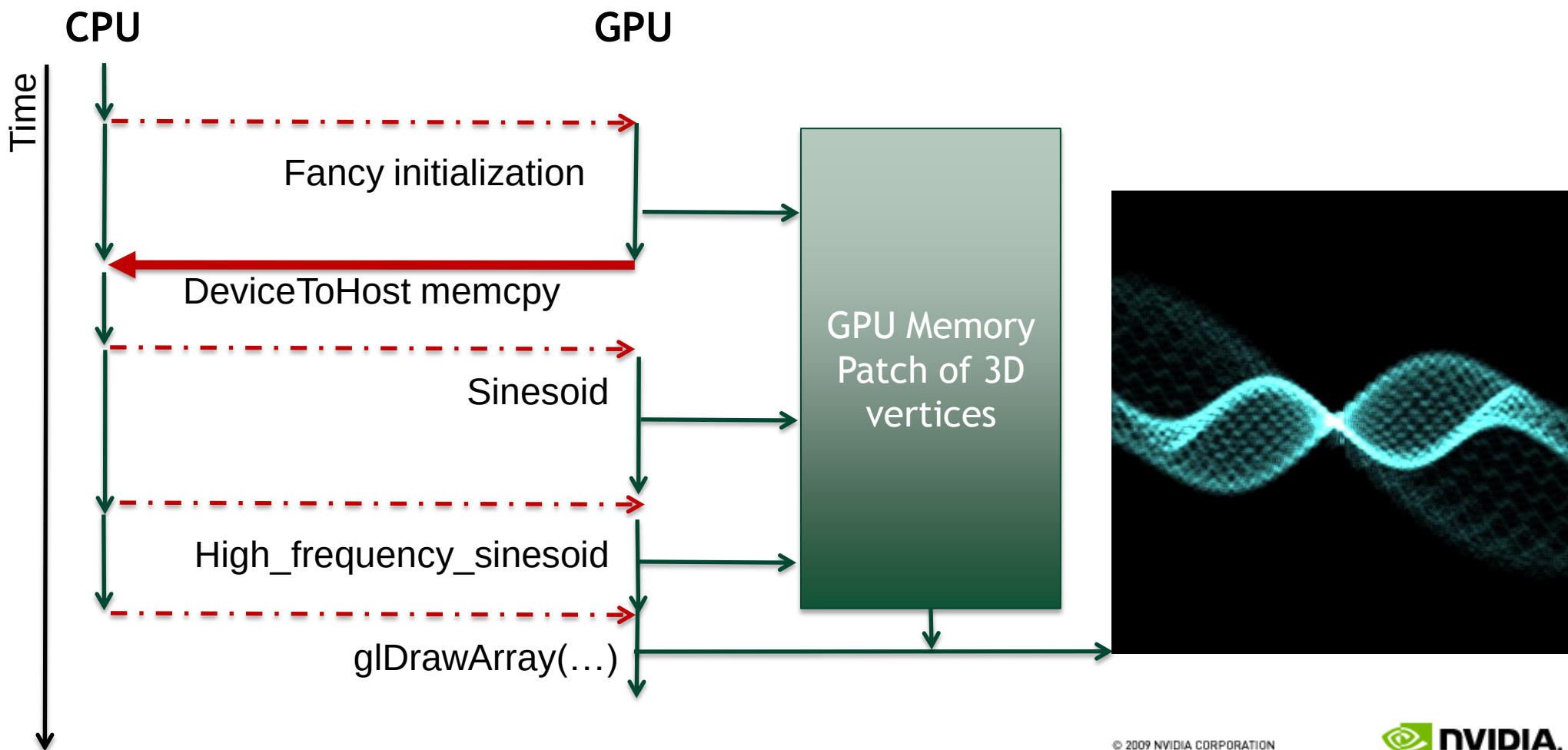
Native GPU Debugging

- No emulation
- No arithmetic discrepancy
- GPU pauses at the breakpoint
- Inspect GPU state and memory
- Faster programmer productivity

CUDA Architecture



My Heterogeneous Computing Sample...



Nexus C-CUDA Debugger Demo

The screenshot displays the Nexus C-CUDA Debugger interface within Microsoft Visual Studio (Administrator). The main window shows the disassembly of `simpleGL_kernel.cu`, with the following code visible:

```
/// @param data data in global memory
// ...
_global__ void init_array(float4 *g_data, unsigned int width, unsigned int height)
{
    unsigned int x = blockIdx.x*blockDim.x + threadIdx.x;
    unsigned int y = blockIdx.y*blockDim.y + threadIdx.y;

    int idx = y*width+x;

    for(int i=0;i<num_iterations;i++)
    {
        g_data[idx].x += (*factor).x; // non-coalesced on purpose, to burn
        g_data[idx].y += (*factor).y; // non-coalesced on purpose, to burn
        g_data[idx].z += (*factor).z; // non-coalesced on purpose, to burn
        g_data[idx].w += (*factor).w; // non-coalesced on purpose, to burn
    }
}
```

The `Locals` pane shows the following variables:

Name	Value	Type
blockDim	{x = 256, y = 1, z = 1}	const dim
gridDim	{x = 1, y = 64, z = 1}	const dim
x	0	unsigned int
y	???	unsigned int
idx	???	int
g_data	0x00110000 {x = -2, y = -0.2528252, z = -2, w = 1}	__device__ shared device
width	256	__device__ shared device
height	64	__device__ shared device

The `Thread` pane shows the following thread information:

Thread	Device	PC	Active Mask
0	Device 0	c:\ProgramData\NVIDIA Corporation\NVIDIA GPU Comput...	0x00000000
1	Device 0	c:\ProgramData\NVIDIA Corporation\NVIDIA GPU Comput...	0x00000000
2	Device 0	c:\ProgramData\NVIDIA Corporation\NVIDIA GPU Comput...	0x00000000
3	Device 0	c:\ProgramData\NVIDIA Corporation\NVIDIA GPU Comput...	0x00000000
4	Device 0	c:\ProgramData\NVIDIA Corporation\NVIDIA GPU Comput...	0x00000000
5	Device 0	c:\ProgramData\NVIDIA Corporation\NVIDIA GPU Comput...	0x00000000
6	Device 0	c:\ProgramData\NVIDIA Corporation\NVIDIA GPU Comput...	0x00000000
7	Device 0	c:\ProgramData\NVIDIA Corporation\NVIDIA GPU Comput...	0x00000000

The `Breakpoints` pane shows the following breakpoints:

Name	Condition	Hit Count
When '0x0011000c' changes (4 bytes)	(no condition)	break always (currently 0)
simpleGL_kernel.cu, line 119	(no condition)	break always
simpleGL_kernel.cu, line 29	(no condition)	break always (currently 0)

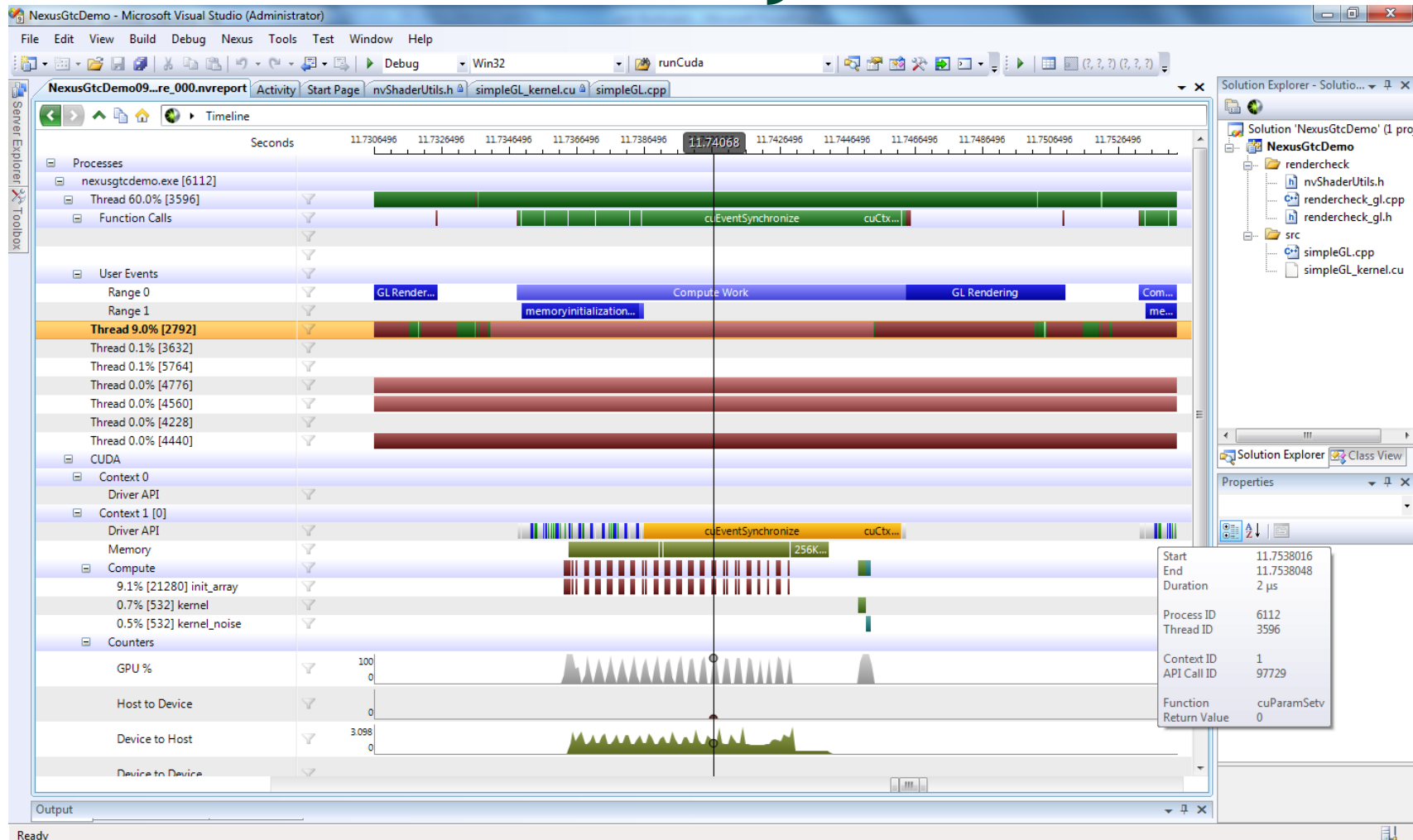
Developing for Heterogeneous Platforms

- CPU and GPU cooperation
- PCI-E / memory transfers
- Latency

Platform Analysis

- Collect platform activity
 - OS - process, thread, thread switch, and module events
 - CPU API Trace - CUDA driver API, DirectX, OpenGL, OpenCL, Cg2.2 and User Events
 - GPU Task Trace - C-CUDA and OpenCL launches and memory copies
- Display summary pages, timeline, API call logs, and GPU task logs

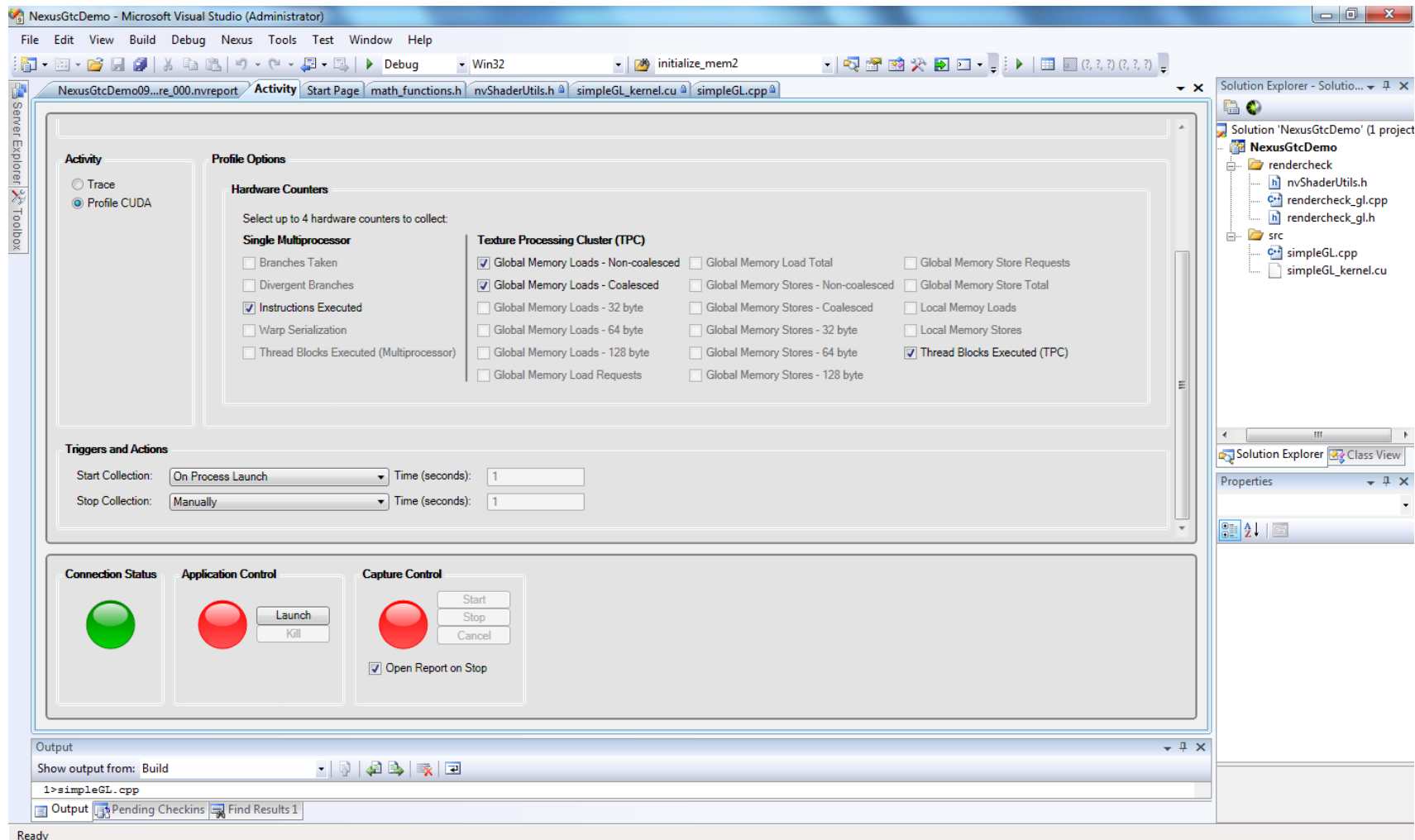
Nexus Platform Analysis Demo



CUDA Profiling

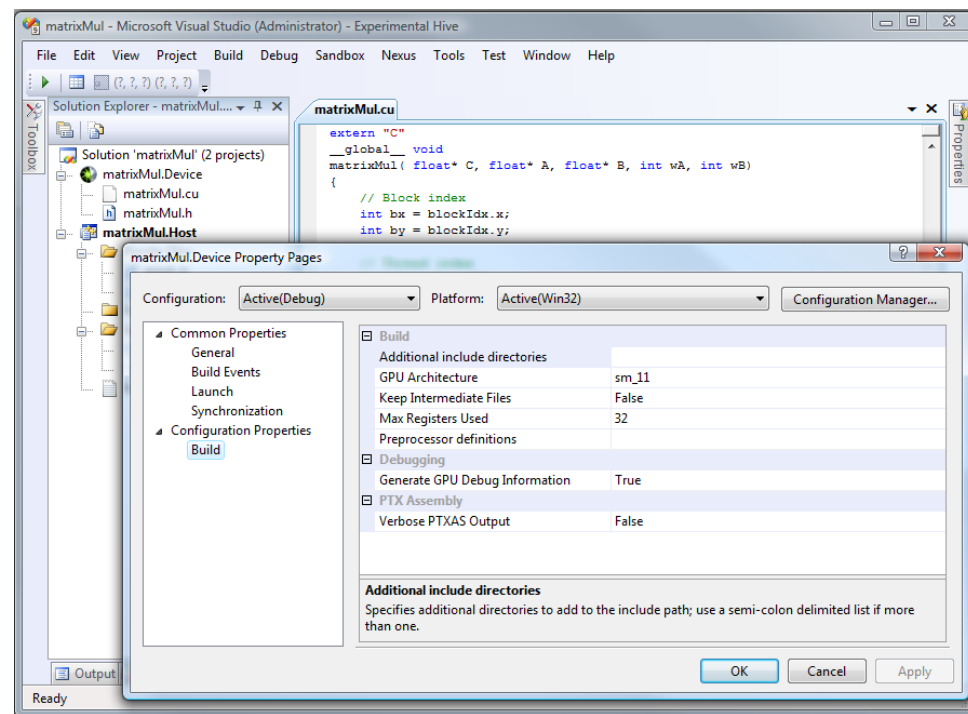
- Kernel tuning basic
 - Memory coalescing
 - Branch divergence
 - Instructions executed
 - Warp serialization
- Use of hardware performance counters

Nexus CUDA Profiler



Nexus Build

- C++ project system
 - Nexus options: launch, sync,...
 - CUDA vsprop files
- CUDA project system
 - NVCC build integration
 - Error reporting
 - Debugger session



Nexus Products

	Nexus Standard	Nexus Professional
Price	Free	\$349
Platforms	Windows Vista SP1 Windows 7	Windows Vista SP1 Windows 7
IDE Integration	Visual Studio 2008 SP1 Standard and above	Visual Studio 2008 SP1 Standard and above
CUDA -C debugging and profiling	✓	✓
DirectCompute debugging and profiling	✓	✓
Remote Debugging	✓	✓
OpenCL Profiling	✓	✓
Memory Checker	✓	✓
Data breakpoints	-	✓
Buffer Visualizer	-	✓
System Trace (CPU + GPU)	-	✓
Priority Ticket Support	-	✓

Supported Operating Systems



Windows Vista SP1

32bit
64bit
32-on-64

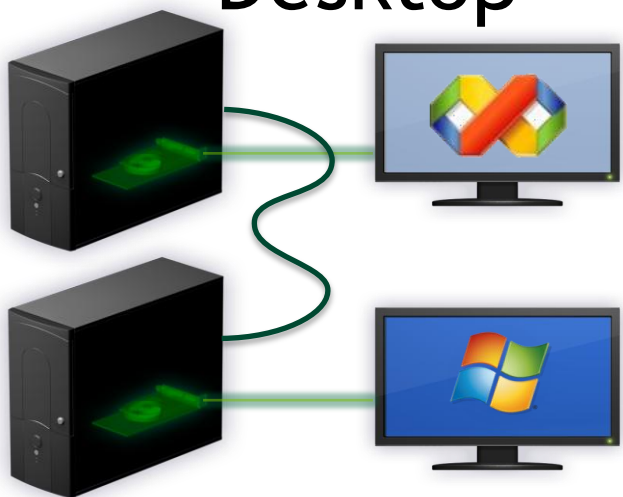


Windows 7

System Requirements

Remote Debugging

Desktop



Mobile



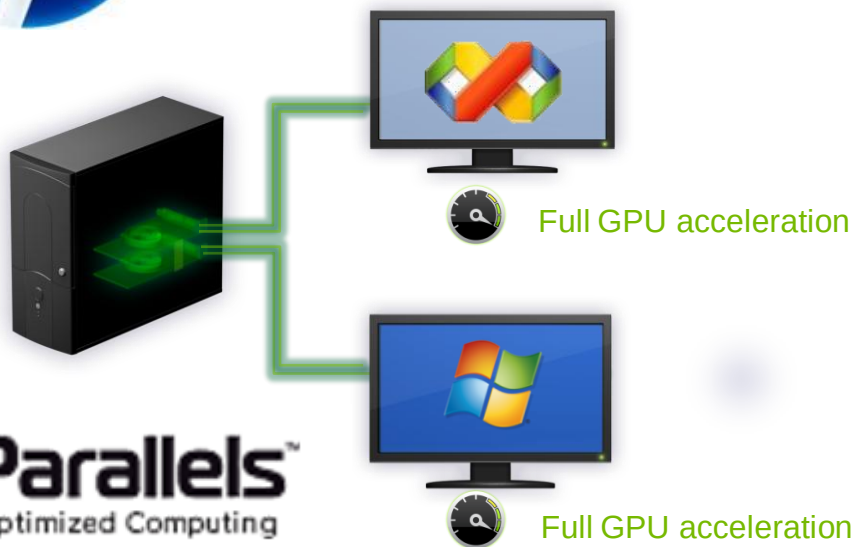
System Requirements



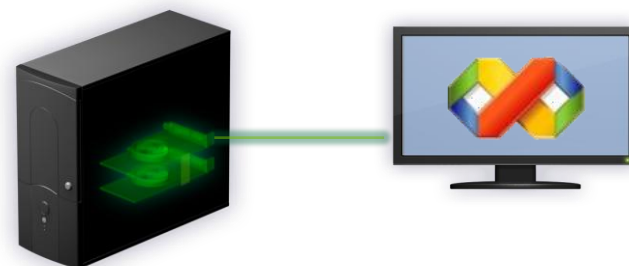
Local Debugging

SLI Multi-OS

Multi-GPU



Parallels™
Optimized Computing



GPU Requirements

- CUDA 1.1 capable GPU
- GeForce 9 and GTX series
- QuadroFX x700 and x800 series
- Tesla C1060
- Fermi architecture-based products
- Non-NVIDIA DirectX 10 and 11 GPUs (with reduced feature set)

Roadmap and Schedule

	Beta1 October '09	Beta2 January '10	Release Q1'10
C-CUDA	✓	✓	✓
DirectX 10	✓	✓	✓
DirectCompute		✓	✓
DirectX 11		✓	✓
OpenCL 1.0		Trace only	Trace only
OpenGL 3.2/Cg2.2	Trace only	Trace only	Trace only

Conclusion and Recap

- Revolutionizing GPU development
 - GPU is now a first-class development target
 - All Compute and Graphics languages and APIs
- Co-Processing Development Solution
 - Clear view of the overall platform activities
 - Deep analysis of specific workloads
- Visual Studio 2008 SP1 Standard and Above
- Windows Vista SP1 and Windows 7

Q&A

- 1-hour Nexus Labs @ Piedmont Room:
 - Friday (2pm-5pm)

- Register for the Beta program today:

<http://developer.nvidia.com/object/nexus.html>

- For Linux - CUDA-gdb talk
 - Friday, Gold Room at 2h30pm

