S3478 - Debugging CUDA Kernel Code with NVIDIA Nsight Visual Studio Edition
Agenda

- What is Nsight and how can it help me?
- Creating projects and CUDA build system - live demo
- Nsight CUDA debugger features helps you find issues - live demo
- Nsight support for CUDA Dynamic Parallelism - live demo
- Summary of Nsight
  - Overview of Nsight System & application trace and CUDA profiling
  - Supported configurations
- Q&A
NVIDIA Nsight Visual Studio Edition
Visual Studio integrated development for GPU and CPU

Build
Debug
Profile
NVIDIA Nsight

GPU Debugger
Local and single GPU 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 and states
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, Direct3D and OpenGL Trace
Trace WDDM and I/O ETW events
Capture call stack and jump to source
NVIDIA Nsight for Compute Developers

**CUDA debugger**
- Debug CUDA kernels directly on GPU hardware
- Info page and Warp watch view
- Use on-target conditional breakpoints to locate

**CUDA memory checker**
- Out of bounds memory access detection
- Enables precise error detection

**Application and system trace**
- Review CUDA activities across CPU and GPU
- Activity correlation panel

**CUDA profiler**
- Source code correlation
- Deep kernel analysis to detect factors limiting maximum performance
- Unlimited experiments on live kernels
CUDA® Build System

- CUDA Toolkit 4.2 and 5.0 support
- Visual Studio 2008 SP1 and 2010 SP1
- CUDA project wizard
- Visual C++ and .NET project integration
- Project setting extensions
NVIDIA Nsight CUDA® Debugging

- Native GPU debugging with mixed CUDA-C/PTX/SASS assembly
- Debug GPU PTX/SASS code without Symbolic info with CUDA-C
- Debugger attach to running process
- On device conditional breakpoint evaluation with program variables
- GPU memory views and data breakpoints
- CUDA expression engine and stack frame support
- Massively-threaded GPU kernels navigation and run-control
- CUDA memory checker
- CUDA Info Tool-Window shows {all CUDA resources | Memory Allocations, Contexts, ...}
Application to debug: Mandelbrot

- Mandelbrot Set is the visual representation of an iterated function on the complex plane
- $Z = Z^2 + C$
Attaching to a running Kernel with Nsight

- Set the Nsight monitor to “Use this monitor for CUDA attach”

- From the command line, enable Nsight to catch GPU exceptions & memory issues:
  - `SET NSIGHT_CUDA_DEBUGGER=2`
  - Setting it to 1 will allow Nsight to catch subset of GPU exceptions
Nsight Debugging

- Make sure “Generate GPU Debug information” is set to “Yes (-G)”
- Debug | Start Debugging (or F5), launches the CPU debugger
- Choose Nsight | Start CUDA Debugging
  - This will launch the “Startup Project”
- Working directory = project directory if setting is empty
  - != CPU directory setting
CUDA info toolwindow

- Provides a view from the CUDA driver API layer, which sits below the CUDA runtime
What is Dynamic Parallelism?

The ability to launch new grids from the GPU
- Dynamically
- Simultaneously
- Independently

Fermi: Only CPU can generate GPU work
Kepler: GPU can generate work for itself
What Does It Mean?

CPU as Co-Processor

GPU

Autonomous, Dynamic Parallelism
Dynamic Work Generation

- Statically assign conservative worst-case grid

- Dynamically assign performance where accuracy is required
CDP example code

```c
__global__ ChildKernel(void *data)
{
    //Operate on data
}

__global__ ParentKernel(void *data)
{
    ...
    ChildKernel<<<16, 1>>>(data);
}

// In Host Code
ParentKernel<<<256, 64>>>(data);

• Recursion is supported, and a kernel may call itself:

__global__ RecursiveKernel(void *data)
{
    if (continueRecursion == true)
        RecursiveKernel<<<64, 16>>>(data);
}
```
Nsight support for Dynamic Parallelism

- CUDA information toolwindow with parent/child relationship:
  - Positive Grid IDs are kernels launched from host
  - Negative Grid IDs are child kernels launched from device

- “Sleeping” status

- Catches invalid pointer arguments to `cudaMemcpy`* ()
Nsight support for Dynamic Parallelism

- Tracks device to device kernel launches
- Active Warp time
- Visual correlation of parent/child in the timeline and graphical call graph
NVIDIA Nsight System Trace

- Application and system trace
  - OS, CPU, Graphics and Compute APIs, Driver and GPU support

- Concurrent kernel execution and memory transfer trace

- Dependency tracking between API and GPU workload

- NVTX API for source code instrumentation
Memory transfers
CUDA runtime and driver calls
CUDA kernel execution
Concurrent kernel execution
Asynchronous memory transfers
Dependency tracking
NVIDIA Nsight CUDA® Profiling

- CUDA profiler with live counter reconfiguration
- Unlimited experiments on live kernels with kernel replay
- Advanced profiling experiments
  - Achieved occupancy
  - Instruction throughput
  - Full memory hierarchy statistics
- Kernel profiling filtering
New in NVIDIA Nsight Visual Studio Edition 3.0

- Kepler 2 architecture support
- CUDA Toolkit 5.0 support
  - Debug and trace kernels using CUDA Dynamic Parallelism (CDP)
  - Debug and profile kernel using CUDA Static Linking
- Ability to debug optimized/release CUDA-C kernels
- Attach debugger to a kernel paused at a breakpoint or exception
- Ability to copy, paste and edit expression in the CUDA warp watch
- Display texture information in CUDA Info page
- Ability to debug GLSL and CUDA GPU code in the same debug session with Maximus™ systems
New in NVIDIA Nsight Visual Studio Edition 3.0

- CUDA Profiler with source code correlation
  - Instruction Count, Divergent Branch, Memory Transactions
  - Annotated Source Viewer for CUDA-C/PTX/SASS
- New CUDA profiler experiments
- System Trace Improvements
  - CDP Trace: Device Launch Trace, Self/Total Active Warp Time
  - CUDA Queue Depth Trace
  - Multi-GPU P2P memory transfers and cudaSetGLDevice
  - File and Disk I/O ETW events
  - WDDM ETW user mode and kernel mode command queues
  - Custom trace data visualization with NVTXT
 Fully Featured Configurations...

- WDDM driver
- Tesla Compute Cluster driver

- Application and system trace
- CUDA profiling
- CUDA debugger
- CUDA memory checker

2 GPUs
Maximus™ Tesla®+GPU
Single GPU
Optimus™ Laptop
Remote PC
How can I learn more about Nsight?

- **Download**
  - [http://www.nvidia.com/nsight](http://www.nvidia.com/nsight)

- **Nsight Visual Studio Edition Developer Forums**

- **Nsight documentation**
  - Nsight | Help | Local (or Online) Help
How can I learn more about Nsight at GTC?

- **Nsight hands-on labs:**
  - Tue 5pm: S3523 - Hands-on Lab: CUDA Application Debugging Using Nsight
  - Thu 5pm: S3535 - Hands-on Lab: CUDA Application Optimization Using Nsight

- **Nsight sessions:**
  - Wed 9am - 10am: S3381 - Developing and Optimizing CPU & GPU Pipelines in Siemens' CUDA Accelerated Solutions
  - Thu 9am - 10am: S3376 - Optimizing Siemens' DirectModel Rendering Engine
  - Thu 10am - 11am: S3377 - Seamless Compute and OpenGL Graphics Development
  - Thu 4pm - 5pm: S3382 - Profiling and Optimizing CUDA Kernel Code

- **Come and visit the Nsight VSE booth at the Exhibition floor:**
  - Tue and Wed from 12pm - 2pm and 6pm - 8pm
  - Thu 12pm - 2pm
Thank you!

- This Session ID: S3478