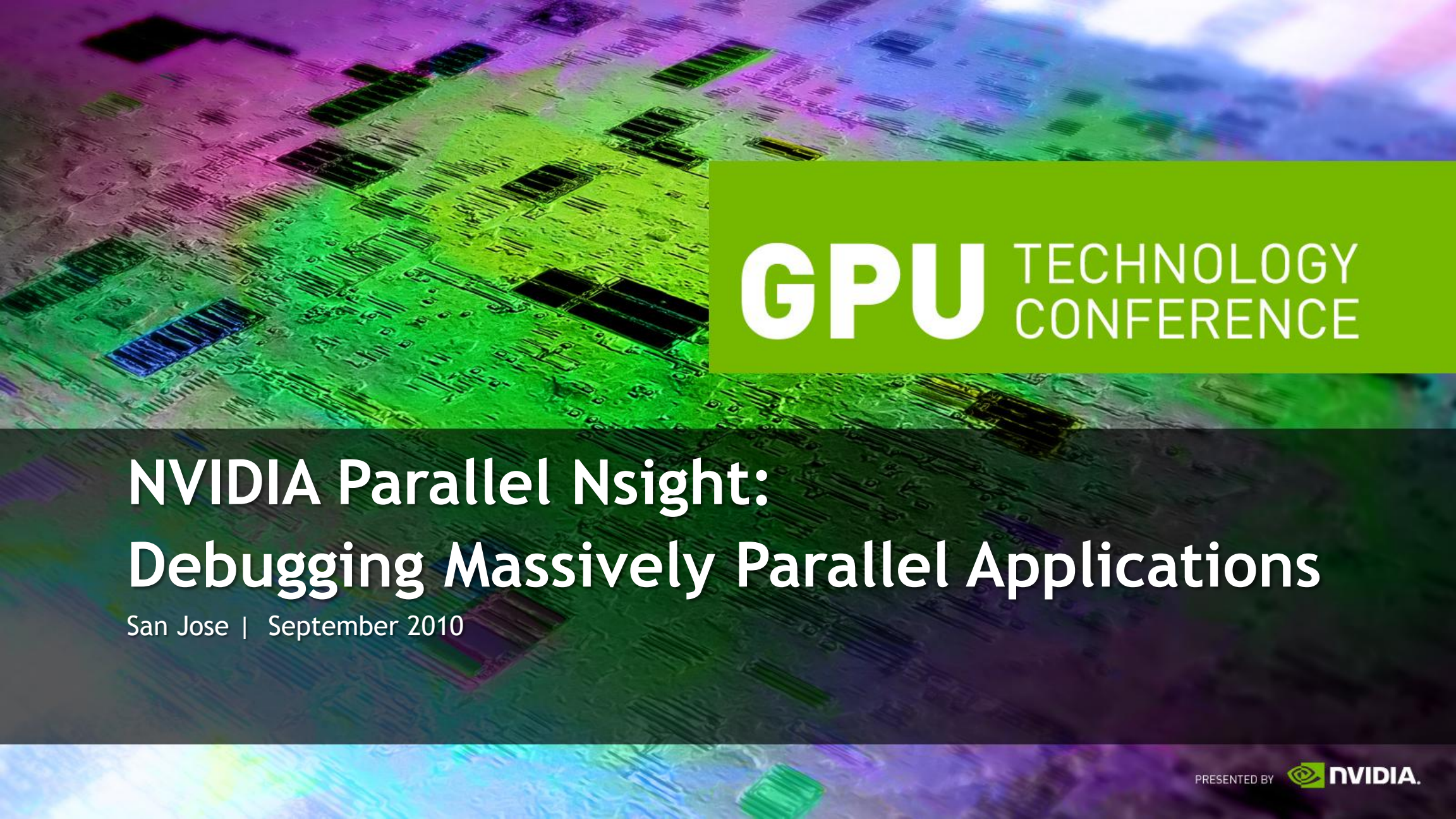


NVIDIA Parallel Nsight: Debugging Massively Parallel Applications

San Jose | September 2010

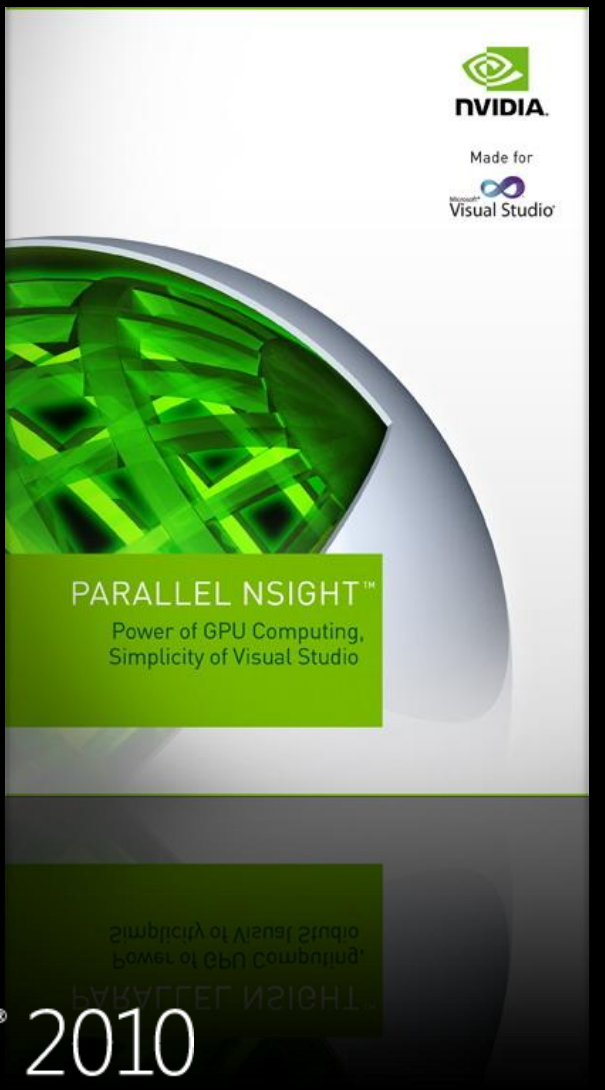
PRESENTED BY NVIDIA.

GPU TECHNOLOGY CONFERENCE

NVIDIA Parallel Nsight™

Application Development Environment for Heterogeneous Platforms

- Build GPU code
- Debug GPU APIs and source code
- Profile GPU APIs and source code
- Trace and analyze GPU+CPU activities



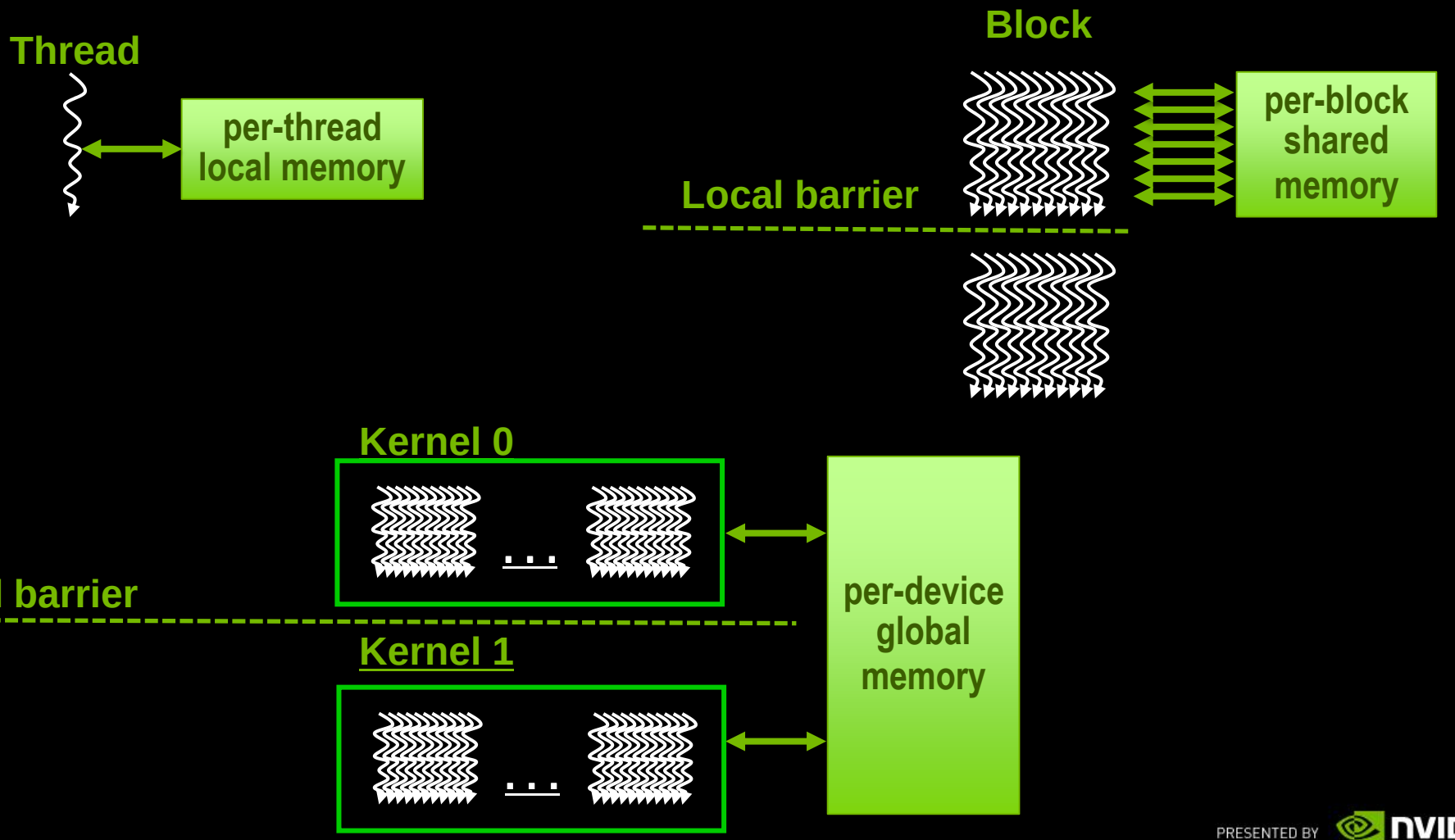
Microsoft® Visual Studio® 2010

PRESENTED BY NVIDIA.

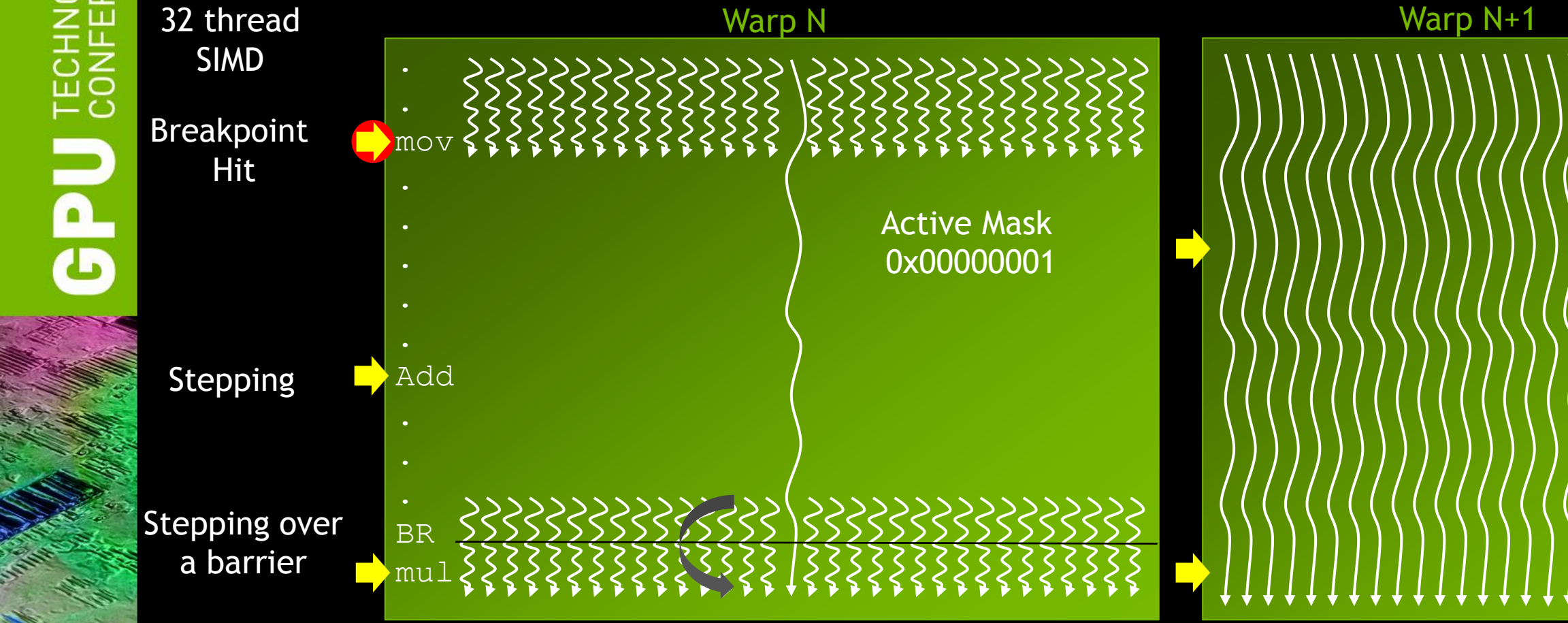
Debugging Massively Parallel Application

- 10000’s threads
- Complex memory Hierarchy
- Warp debugging
- Synchronization/barrier

CUDA Architecture



Warp Debugging

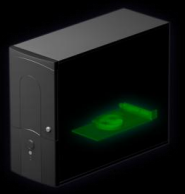


Parallel Nsight - Debugger

- CUDA-C kernels
- DirectCompute shaders
- HLSL graphics shaders
- Conditional breakpoint
- Expression engine
- Data breakpoint
- GPU memory viewer

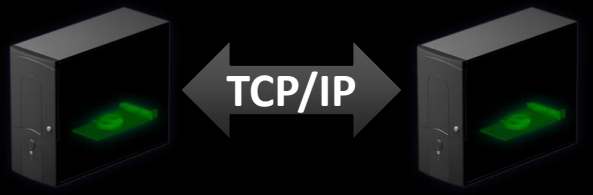
Hardware Configurations

Single machine, Single GPU



Analysis
Graphics Inspector

Two machines, connected over the network



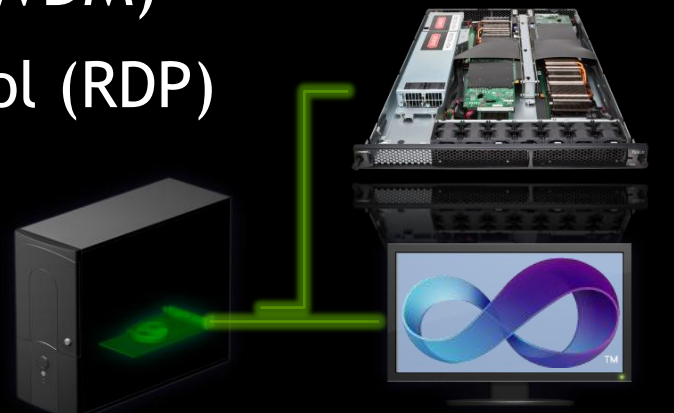
Debugger
Analyzer
Graphics Inspector

SLI or SLI MultiOS: Single machine, Dual GPU

Debugger
Analyzer
Graphics Inspector

Tesla Compute Cluster (TCC) Support

- GPU registered as a compute device (WDM)
- Enables GPUs for Remote Desktop Protocol (RDP)
- Delivers increased performance



- Supported on Windows HPC Server 2008 R2 and Windows 7
- Tesla S/C/M series

Nsight GTC 2010 Demo Introduction

CPU

<<<sobel>>>

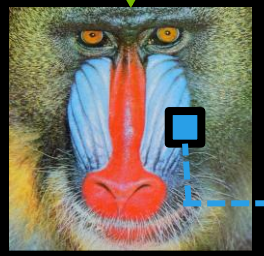
<<<cubegen>>>

glDrawElement(...)

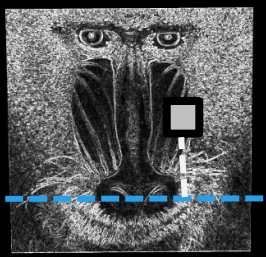
PCI-E

t

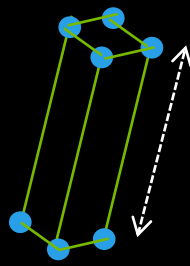
GPU



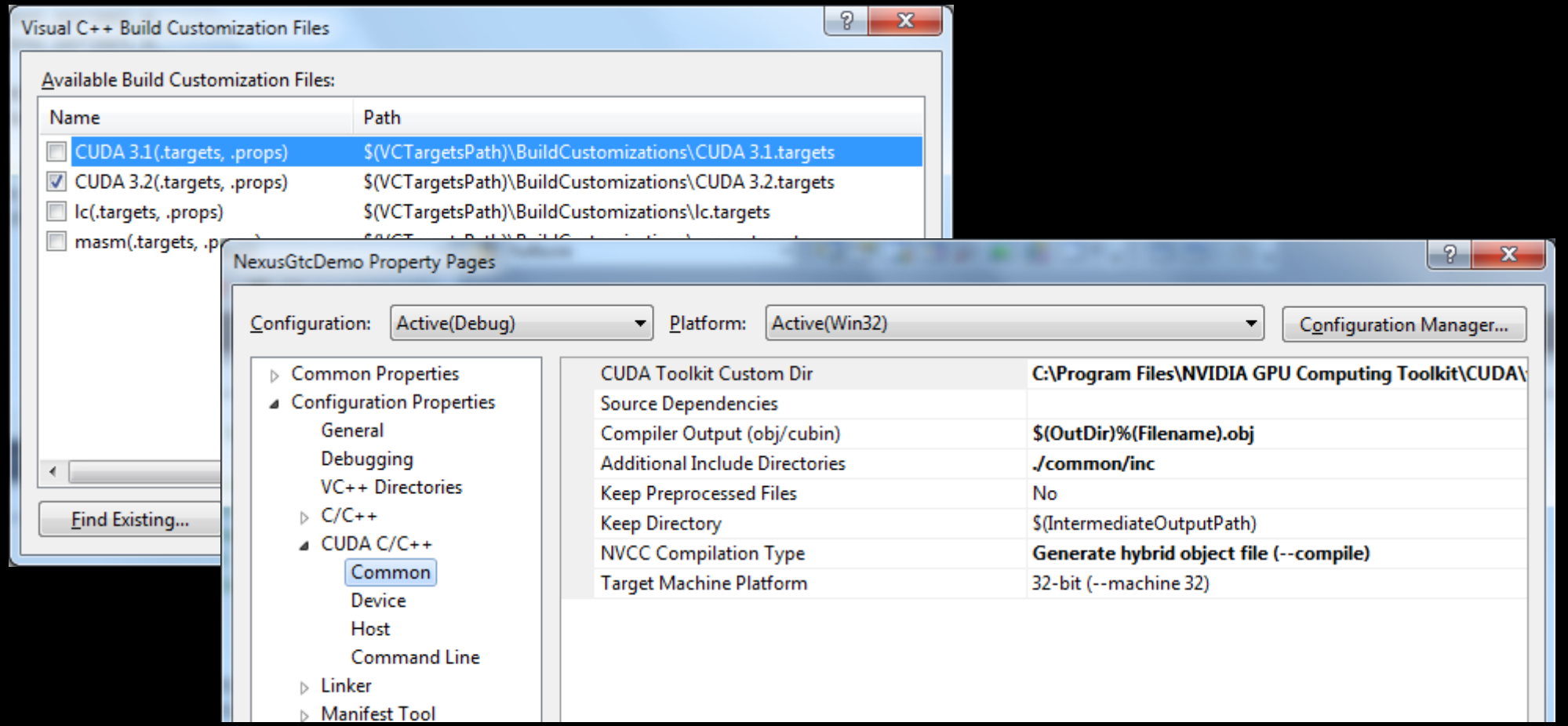
sobel



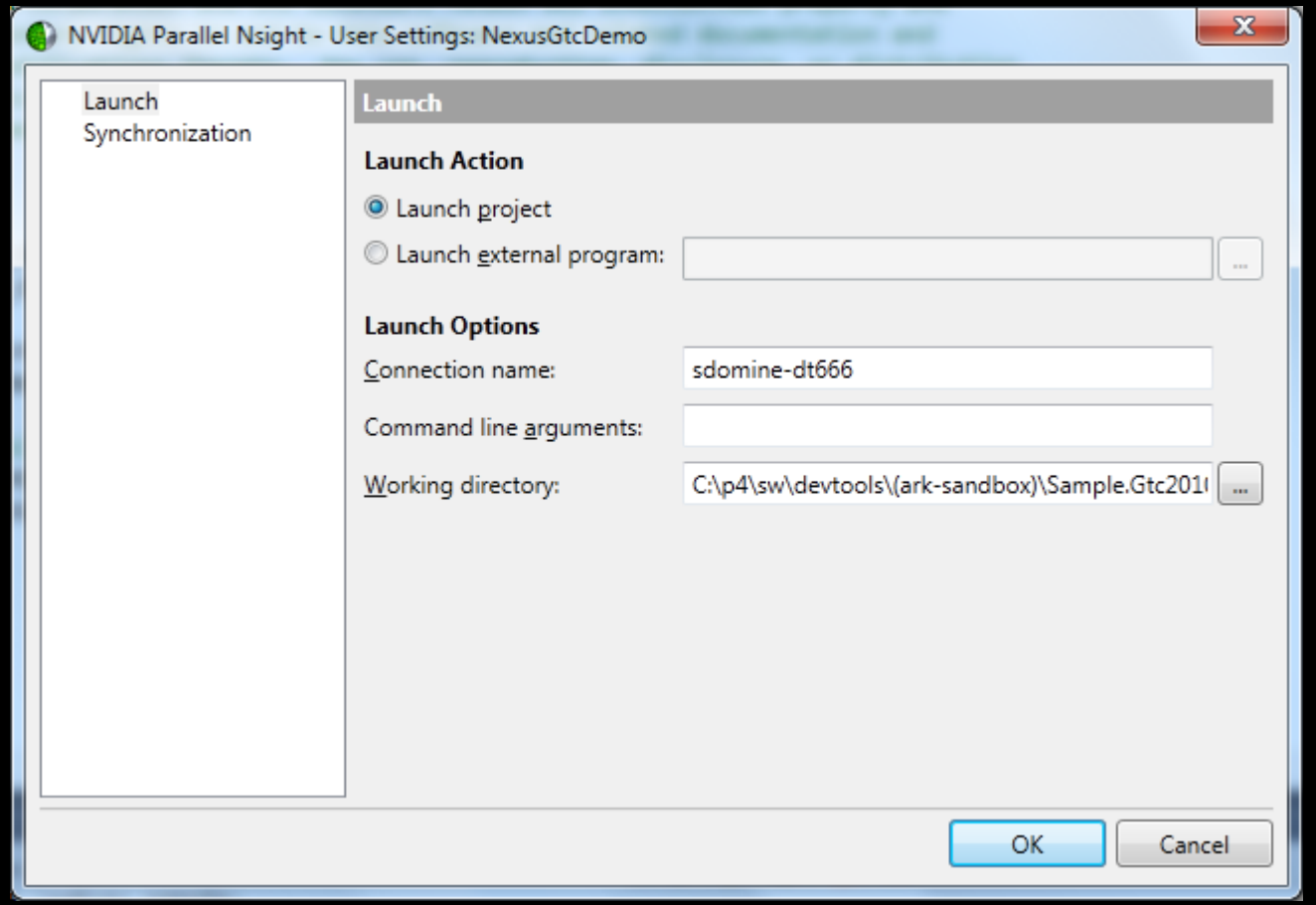
cubegen

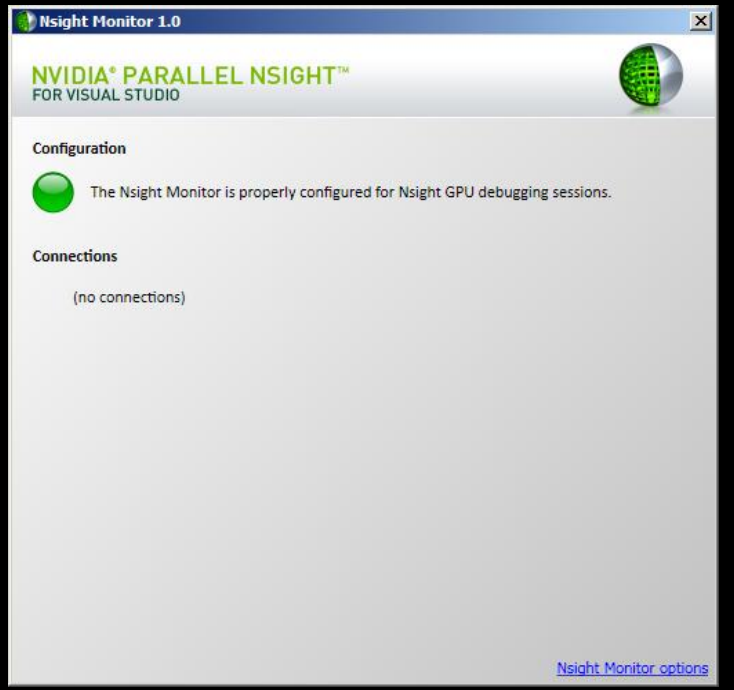


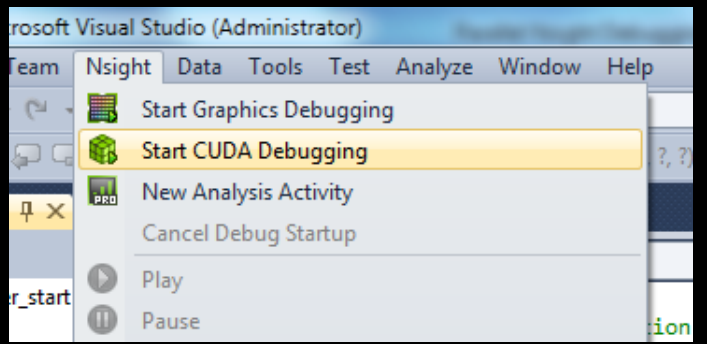
Setting up CUDA builds in VS2010



Remote Debugging







CUDA Memory Checker

```
//
            {
                x = i-1;
                offset = (x - w) * 3;
                color = make_float3(float(pCudaTexPtr[offset + 2]), float(pCudaTexPtr[offset+1]), float(pCudaTexPtr[offset]));
                pix00 = (color.x + color.y + color.z) / 3;
            }
```

100 %

Output

Show output from: Nsight

```
CUDA Memory Checker detected 1 threads caused an access violation:
Launch Parameters
    CUcontext    = 07a58108
    CUstream     = 07dcc830
    CUmodule     = 07dccaf0
    CUfunction   = 07dcd7a0
    FunctionName = _Z8SobelTexPhS_jiif
    gridDim      = {512,1,1}
    blockDim     = {384,1,1}
    sharedSize   = 0
    Parameters:
    Parameters (raw):
         0x05200000 0x05300000 0x00000200 0x00000200
         0x00000200 0x3f800000
GPU State:
   Address  Size      Type       Block  Thread         blockIdx  threadIdx      PC  Source
-------------------------------------------------------------------------------------------
  051fffff     1    adr ld           1       0         {1,0,0}    {0,0,0}  000590  c:\p4\sw\devtools\(ark-sandbox)\sample.gtc2010.demo\sobel
```

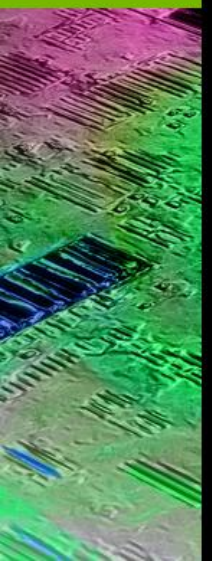
Sobel Filter Debugging

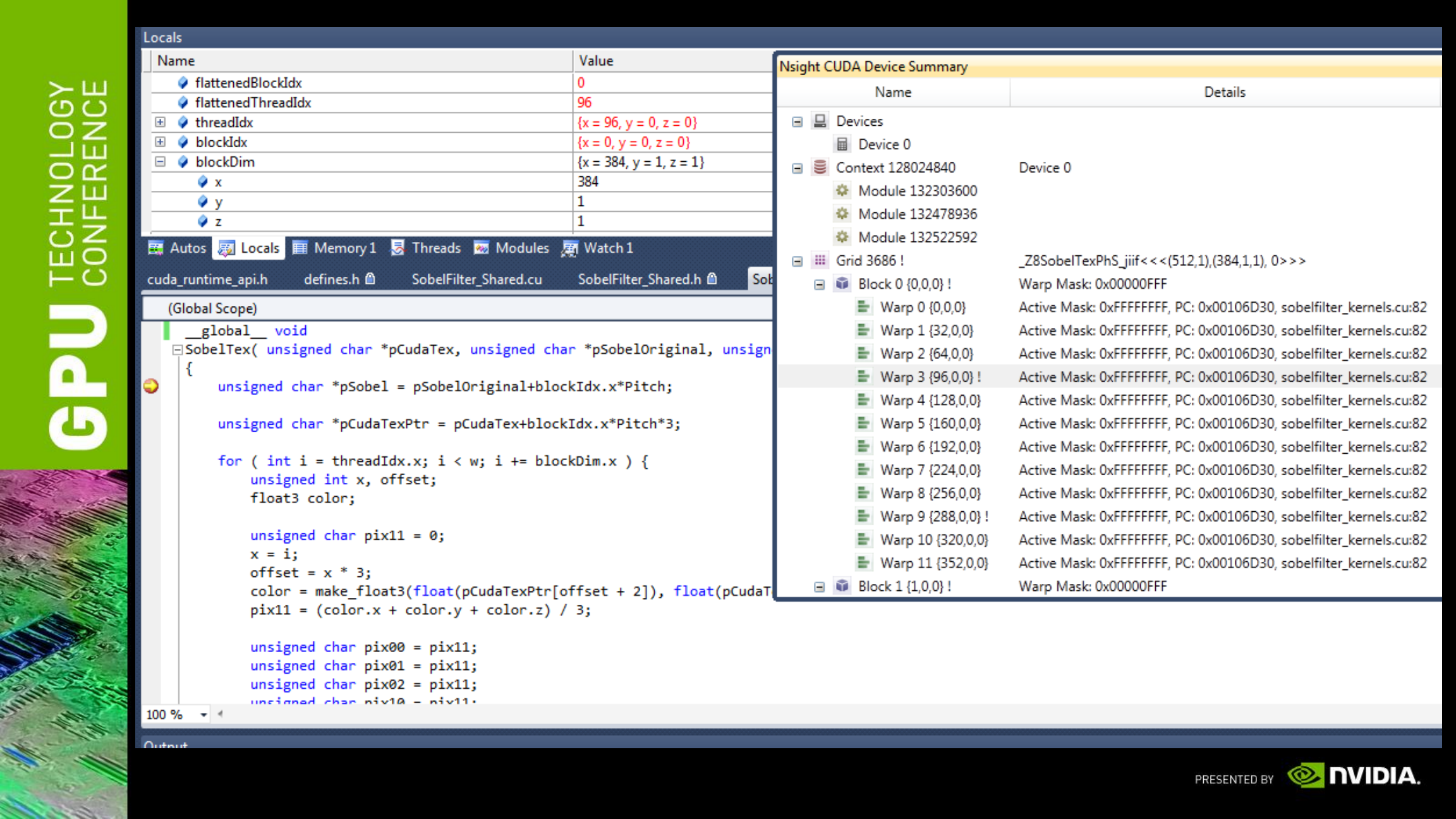
$$C_{horizontal} = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

$$C_{vertical} = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$$

$$Magnitude_{Sobel} = norm \bullet \sqrt{G_{horizontal}^2 + G_{vertical}^2}$$

$$Direction_{Sobel} = \arctan\left(\frac{G_{vertical}}{G_{horizontal}}\right)$$




GPU TECHNOLOGY CONFERENCE
Locals
Name
Value
flattenedBlockIdx
0
flattenedThreadIdx
96
threadIdx
{x = 96, y = 0, z = 0}
blockIdx
{x = 0, y = 0, z = 0}
blockDim
{x = 384, y = 1, z = 1}
x
384
y
1
z
1
Autos
Locals
Memory 1
Threads
Modules
Watch 1
cuda_runtime_api.h
defines.h
SobelFilter_Shared.cu
SobelFilter_Shared.h
(Global Scope)
__global__ void
SobelTex(unsigned char *pCudaTex, unsigned char *pSobelOriginal, unsign
{
unsigned char *pSobel = pSobelOriginal+blockIdx.x*Pitch;
unsigned char *pCudaTexPtr = pCudaTex+blockIdx.x*Pitch*3;
for (int i = threadIdx.x; i < w; i += blockDim.x) {
unsigned int x, offset;
float3 color;
unsigned char pix11 = 0;
x = i;
offset = x * 3;
color = make_float3(float(pCudaTexPtr[offset + 2]), float(pCudaT
pix11 = (color.x + color.y + color.z) / 3;
unsigned char pix00 = pix11;
unsigned char pix01 = pix11;
unsigned char pix02 = pix11;
100 %
Output
Nsight CUDA Device Summary
Name
Details
Devices
Device 0
Context 128024840
Device 0
Module 132303600
Module 132478936
Module 132522592
Grid 3686 !
_Z8SobelTexPhS_jiif<<<(512,1),(384,1,1), 0>>>
Block 0 {0,0,0} !
Warp Mask: 0x00000FFF
Warp 0 {0,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 1 {32,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 2 {64,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 3 {96,0,0} !
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 4 {128,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 5 {160,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 6 {192,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 7 {224,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 8 {256,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 9 {288,0,0} !
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 10 {320,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Warp 11 {352,0,0}
Active Mask: 0xFFFFFFFF, PC: 0x00106D30, sobelfilter_kernels.cu:82
Block 1 {1,0,0} !
Warp Mask: 0x00000FFF
PRESENTED BY
NVIDIA.

Memory 1

Address: &pSobel[i] Columns: Auto

```
0x0000000005300060  d2 bc 86 5c 3e 52 70 86 a6 b0 b0 a6 90 92 92 88 6c 60 5c 4c 4c 3c 14 d4 0e 52 10 3e 00 12 6a 6a 00 00 00  Ò..\>Rp.¦°°¦.’’ˆl`\LL<.
0x0000000005300083  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 7a 7a 70 6e 6e 6e  ..........................
0x00000000053000A6  38 16 48 5c 5c 60 64 96 b4 b4 82 64 64 68 94 94 7e 68 76 76 88 88 72 88 88 88 00 00 00 00 00 00 00 00  8.H\\`d–´´.ddh””~hvvˆˆr
0x00000000053000C9  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..........................
0x00000000053000EC  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ..........................
0x000000000530010F  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 74 74 76 78 62 62 82 76 76 96 8c 94 94 76 76 94 a0 a0  .................ttvxbb
0x0000000005300132  a0 a0 a0 a0 94 76 80 80 6e 4c 34 4c 76 76 76 64 6e 6e 6e 64 50 78 78 38 6a 6a 66 62 78 78 78 50 78 78 6e      ”v€€nL4Lvvvdnnnd Pxx
0x0000000005300155  6a 6a 6a 78 66 24 28 06 06 34 08 1c ea 56 34 34 1e 40 40 fe ae 98 80 94 aa c0 7e 7e 3a 7c cc 24 0c b8 84  jjjxf$(..4..êV44.@@þ®˜€
```

Autos | Locals | Memory 1 | Threads | Modules | Watch 1

cuda_runtime_api.h | defines.h | SobelFilter_Shared.cu | SobelFilter_Shared.h | SobelFilter_kernels.cu | simpleGL.cpp | simpleGL_kernel_fixed.cu

(Global Scope) SobelTex(unsigned char * pCudaTex, unsigned char * pSobelOriginal, unsigned int

```
            offset = (x + w) * 3;
            color = make_float3(float(pCudaTexPtr[offset + 2]), float(pCudaTexPtr[offset+1]), float(pCudaTexPtr[offset]));
            pix21 = (color.x + color.y + color.z) / 3;

            if (i < w - 1)
            {
                x = i+1;
                offset = (x + w) * 3;
                color = make_float3(float(pCudaTexPtr[offset + 2]), float(pCudaTexPtr[offset+1]), float(pCudaTexPtr[offset]));
                pix22 = (color.x + color.y + color.z) / 3;
            }
        }

        pSobel[i] = ComputeSobel(pix00, pix01, pix02,
                                 pix10, pix11, pix12,
                                 pix20, pix21, pix22, fScale );
    }
}

// Wrapper for the __global__ call that sets up the texture and threads
extern "C" void sobelFilter(unsigned char * cuda_tex, unsigned char *sobel_tex, int iw, int ih, enum SobelMode mode, float fScale, cudaStream_t
```

100 %

Locals

Name	Value	Type
gridDim	{x = 512, y = 1, z = 1}	const dim3
ul	16	unsigned char
um	16	unsigned char
ur	16	unsigned char
ml	16	unsigned char
mm	???	unsigned char
mr	16	unsigned char
ll	77	unsigned char

Autos | Locals | Memory 1 | Threads | Modules | Watch 1

cuda_runtime_api.h | defines.h | SobelFilter_Shared.cu | SobelFilter_Shared.h | SobelFilter_kernels.cu | simpleGL.cpp | simpleGL_kernel_fixed.cu

(Global Scope) — ComputeSobel(unsigned char ul, unsigned char um, unsigned char ur, unsi

```
__device__ unsigned char
ComputeSobel(unsigned char ul, // upper left
             unsigned char um, // upper middle
             unsigned char ur, // upper right
             unsigned char ml, // middle left
             unsigned char mm, // middle (unused)
             unsigned char mr, // middle right
             unsigned char ll, // lower left
             unsigned char lm, // lower middle
             unsigned char lr, // lower right
             float fScale )
{
    short Horz = ur + 2*mr + lr - ul - 2*ml - ll;
    short Vert = ul + 2*um + ur - ll - 2*lm - lr;
    short Sum = (short) (fScale*(abs(Horz)+abs(Vert)));
    //
    return (unsigned char) Sum;
}

// cuda tex - BGR image
```

100 %

CUDA Memory Breakpoint

```
{
    unsigned int x = blockIdx.x*blockDim.x + threadIdx.x;
    unsigned int y = blockIdx.y*blockDim.y + threadIdx.y;

    int idx = y*width+x;
    if (idx < width && idx % 2 == 0)
        g_data[idx] = 255;
}

// Wrapper for the __global__ call that sets up the texture and threads
extern "C" void sobelFilter(unsigned char * cuda_tex, unsigned char *sobel_tex, int iw, int ih, enum SobelMode mode, float fScale, cudaS
```

100 %

Breakpoints

New | Columns | Search: | In Column: All visible

Name	Labels	Condition	Hit Count
SobelFilter_kernels.cu, line 172		(no condition)	break always (currently 0)
When '0x0000000005200000' changes (4 bytes)		(no condition)	break always (currently 0)

Autos | Locals | Memory 1 | Threads | Modules | Watch 1 | Call Stack | Breakpoints | Output

Ln 185 Col 1

Schedules and Where To Get It…

- Parallel Nsight 1.5 RC available now
- Parallel Nsight 1.5 Final soon after GTC

www.nvidia.com/ParallelNsight

Conclusion and Q&A

- GPU is a first-class development target
- CUDA-C development in Visual Studio 2010
- Tesla Compute Cluster support for Professionals
- CUDA Toolkit 3.1 and 3.2 Support

GPU TECHNOLOGY CONFERENCE

NVIDIA Parallel Nsight™ at GTC 2010

Parallel Nsight Lounge by Microsoft (Ballroom Concourse)

From 10am-8pm each day, give-a-ways daily at 3pm

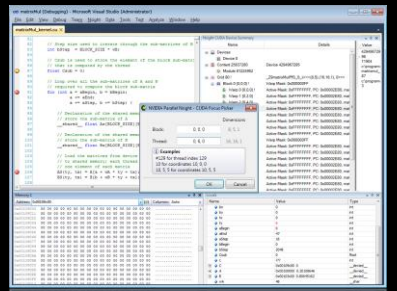
www.nvidia.com/ParallelNsight

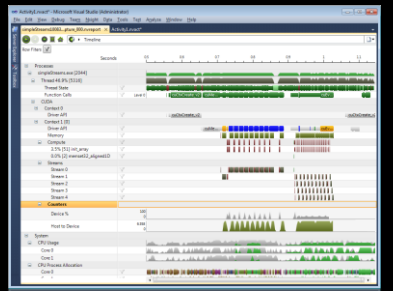
All Parallel Nsight Sessions at GTC are in Room B

Tues, 3-3:50pm: GPGPU Development for Windows HPC Server [Microsoft]

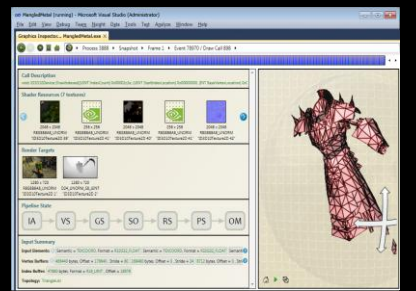
Tues, 4-4:50pm: Parallel Nsight: Analyzing Massively Parallel Applications

Tues, 5-5:50pm: Parallel Nsight for Accelerated DirectX 11 Development


Compute Debugger


System Analysis


Graphics Debugger


Graphics Inspector