GPU TECHNOLOGY CONFERENCE

Developing Next-Generation CUDA Acceleration in Wolfram's *Mathematica* with NVIDIA® Nsight[™] Visual Studio Edition

Abdul Dakkak Kernel Developer Wolfram Research

Sébastien Dominé Sr. Dir. Developer Tools NVIDIA

Agenda

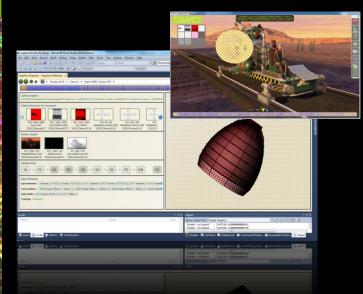
- Nsight Visual Studio Edition Update
- CUDA Development in Wolfram's Mathematica
- The future of Nsight Visual Studio Edition
- Conclusion

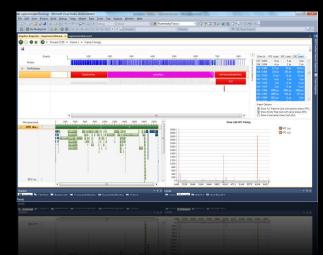
NVIDIA® Nsight[™] Visual Studio Edition Software Development Platform for GPU and CPU

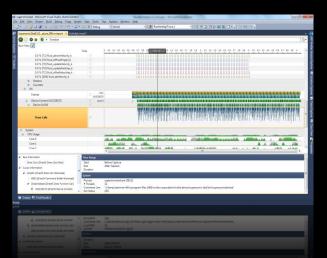


Nsight Visual Studio Edition for Graphics Developers

- Frame debugger for Direct3D
- HLSL Shader debugger
- Frame profiler for Direct3D
- Application and system trace

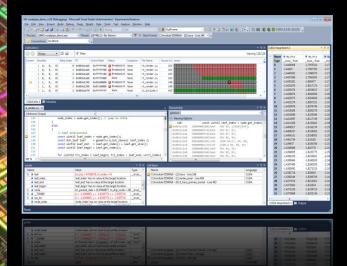


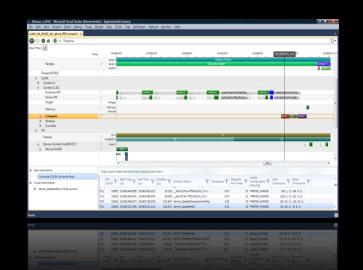


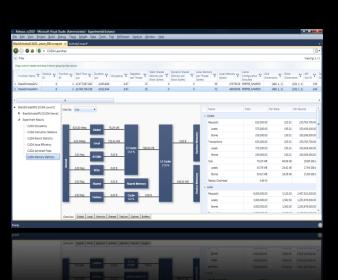


Nsight Visual Studio Edition for CUDA® Developers

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







New in Nsight Visual Studio Edition 2.2

- Kepler architecture support
- Fully featured on single GPU systems
 - Local CUDA Debugging
 - CUDA memory checker
- Warp freeze/thaw run-control
- Debug Kernel without symbols SASS and PTX
- New Analysis summary page



<u>A</u> ddress:	0x0001	db78
👽 Viewir	ng Optio	ins
0x0001	Ldb28	@P0 BRA 0x738; # Target=0x0001db38
0x0001	Ldb30	STS.S [R25], RZ;
0x0001	Ldb38	MOV R4, R21;
0x0001	Ldb40	MOV R7, R24;
0x0001	Ldb48	CAL 0x1b88; # Target=0x0001ef88
0x0001	Ldb50	STS.S [R25], R4;
0x0001	Ldb58	IADD R19, R19, c[0x0][0x8];
0x0001	Ldb60	ISETP.LT.U32.AND P0, pt, R19, R18, pt;
0x0001	Ldb68	@P0 BRA 0x440; # Target=0x0001d840
0x0001	Ldb70	IMAD.U32.U32.S RZ, R1, RZ, RZ;
0x0001	Ldb78	BAR.RED.POPC RZ, RZ;
0x0001	Ldb80	LD R4, [R2+0x420c];
0x0001	Ldb88	LD R5, [R2+0x4204];
0x0001	Ldb90	LD R0, [R2+0x4208];
0x0001	Ldb98	LD R3, [R2+0x4200];
0x0001	Ldba0	SSY 0x18b0;
0x0001	Ldba8	FMNMX R25, R5, R4, !pt;
🔶 0x0001	Ldbb0	FMNMX R26, R3, R0, !pt;
0x0001	Ldbb8	F2F.CEIL R0, R25;

What is *Mathematica*?

- Computational Engine
- Functional, pattern based, procedural, ... everything is data
- Covers Many Domains All in One Package no need for extensions and toolkits

Structure of Mathematica Code Base

- Mainly written in C and Mathematica
- Some bits are written in Java
- A separation between the frontend and the kernel into different processes
- Uses and loads external libraries on demand for certain operations
- Has a linking mechanism that allows loading C, Java, .Net, and Python code into the system
- CUDALink is one of those linking mechanisms

Mathematica Frontend and Kernel

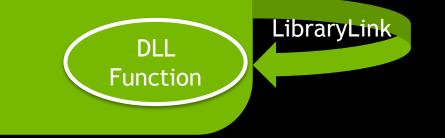
Mathematica Kernel All computation happens in the kernel



Mathematica Frontend All GUI and visualization happens here

Mathematica Linking Mechanism

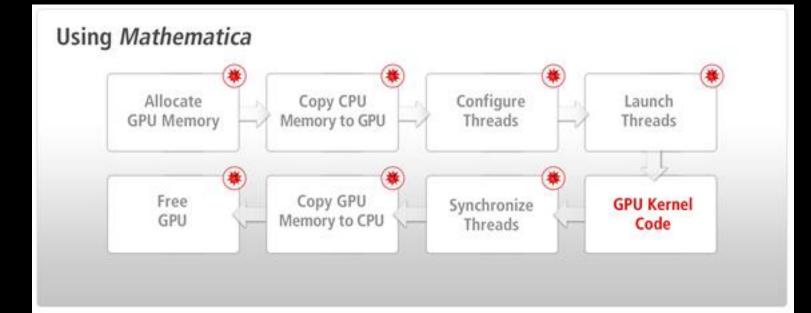




What is CUDALink?

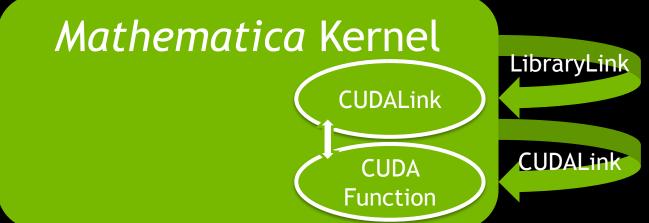
- A way to load CUDA programs into *Mathematica*
- Handles all the trivial, repetitive, and sometime error prone host code that developers have to write
- Allows CUDA programs to benefit from *Mathematica*'s features and vice versa





Structure of CUDALink

- Written in C, CUDA, and *Mathematica*
- Uses NVIDIA libraries such as CUBLAS, CUFFT, CURAND, and Thrust
- Loaded as an external library by the main Mathematica kernel



Demo: CUDALink Usage

Challenges While Developing CUDALink

- Initially, lack of Nsight and printf
- The initial hardware requirements for Nsight were not easy to configure (QA, for example, did not have this setup)
- Large code base means that you cannot debug the entire CUDA code base

Demo: Attaching the CUDA Debugger to a *Mathematica* Process

Demo: Profiling the CUDA Code with Nsight

Previous Development Workflow

- Write implementation in C
- Port implementation into CUDA
- Scratch head if something does not work
- Optimize based on knowledge of the hardware
- Write tests to make sure things continue to work

Current Development Workflow

- Write implementation in CUDA
- Debug CUDA implementation
- Memory check CUDA implementation
- Optimize the implementation based on what the profiler tells you

Pitfalls and Experiences

- A small change in the way we develop CUDA code the "old school" way of debugging and profiling
- Initial setup challenges were answered by the documentation pages
- Enhanced and accelerated our development and QA process

Wish List

- Hard to debug large CUDA files
- Live variable range is short during debugging
- As CUDA projects get larger: the compilation process gets slower, debugging get harder, and the release binary size gets larger
- Run static analysis without writing the host code
- Attach debugger to a CUDA function without host code symbols

The Future for Nsight Visual Studio Edition

CUDA 5.0 and dynamic kernel debugging

- Support for debugging GPU object linking
- Performance Bottleneck w/ Source code correlation

Kernel Performance Limiter Analysis

System trace File I/O

Profiling with source code correlation

LOGY

_																	
	tesselateNURBSPatches_Kernel		Grid Dim:	{100, 1, 1}					Bloc	k Dim: {512, 1, 1}			Duration: 14	06.4 µs			
File:	NURBSTessellation4.cu View: Source	e and :	SASS	• 🔳 🗄	High to Low: 🏼	🛓 🔹 Low to Higi	h: 🚽	۰.									
Line	Source		Instructions Executed	tions Thread Instructions Thread		Line				Instructions Executed	Thread Instructions V Executed	Thread Instructions Executed (False)	Thread Instructions Predicated Off	7			
88	//return the index to the knot span that u	u is	co						931	/*1D10*/ @P0	BRA 0x1ac8; # Targ	et=0x00001ac8	1600	51200	51200	100.	0
89	device unsigned int findSpan(float u,	-							932	/*1D18*/	EXIT;		1600	51200) O	0.	.0
90	{								933	/*1D20*/	IADD R5, R5, 0x1;		1200	36300) o	0.	0
91	<pre>for(unsigned int i = 0; i < nKnots - 1;</pre>	i++)	22800	689700	96800	14.0	1		934	/*1D28*/	MOV R9, RZ;		1200	36300) o	0.	0
92	{								935	/*1D30*/	ISETP.GE.U32.AND P	0, PT, R5, R4, PT;	1200	36300) O	0.	0 = 7
93	if(u >= ku[i] && u < ku[i+1])	if(u >= ku[i] && u < ku[i+1])			169400	19.2	-		936 /*1D38*/ 0				1200	36300	36300	100.	0
94	return i;						938 939			-	RA 0x11f8; # Target=0x000011f8		36300) C	0.	0	
95	}									IADD R5, R5, 0x1; MOV R3, RZ; ISETP.GE.U32.AND P0, PT, R5, R4, PT;		1200	36300) O	0.	0	
96	return 0;											1200	36300) C	0.	0 🔳	
97	}											1200	36300	0 0	0.	•	
98									/*1D60*/ @PO			1200	36300	36300	100.	0	
99	global void tesselateNURBSPatches_Kern	NUR								BRA 0x1160; # Target=0x00001160		1200	36300	0 0	0.	0	
100	{								/*1D70*/	IADD R1, R1, -0x28		34000	202800) C		_	
101	NURBSPatch *patch = &patches[blockIdx.x]	1600	51200	0	0.0	-		944	/*1D78*/	S2R R0, SR_LMemHi0	ff;	34000	202800	0 0	0.	.0 💽 🗕	
⊿— te	sselateNURBSPatches_Kernel [CUDA Launch]	Drag	g a column header and drop	it here to group by t	hat column												
1	- tesselateNURBSPatches_Kernel [CUDA Kernel]	F	File 🛛 🝸	Line # 🍸 🛛 Instru	ctions Executed 🍸	Thread Instruction	ons l	Execut	ed	Thread Instruct	tions Executed (False) 🏹	Thread Instructions Pred	dicated Off 🍸 🗸	Active Mask		Predicates	
	Experiment Results	1 n	nurbstessellation4.cu 🏲	66	4720	5			17820	00	0		0.0				
	 CUDA Occupancy CUDA Code Correlation 		nurbstessellation4.cu 🎙		4720	0			17820	00	0		0.0			=	
			nurbstessellation4.cu 🎙		3400	5	202		20280	00	0					=	
	CUDA Instruction Count		nurbstessellation4.cu 🎙		3400	0			20280	00	0					1	
			nurbstessellation4.cu 🎙		3400	5			20280	00	0					1	
			nurbstessellation4.cu 🎙		3400	2			20280	00	0						
5			nurbstessellation4.cu 🄻		3400	5			20280	00	0			I.III. I.B			-
		٠ [•

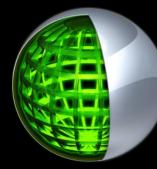
Conclusion

- Full-featured CUDA development on a single GPU
- Advanced CUDA debugging with more control
 - Attach to process
 - CUDA Info and parallel warp pages
- System trace for finding where to focus optimization effort
- Powerful profiling experiments for accurate performance characterization

Mathematica Presentations

- Mathematica as a Practical Platform for GPU-Accelerated Finance: Wednesday 5:00PM (S0100)
- GPU Based Numerical Methods in Mathematica: Thursday 14:30 (S0106)

Nsight Visual Studio Edition@GTC'12



- Download
 - http://developer.nvidia.com/nvidia-nsight
- NVIDIA Nsight Visual Studio Edition Trainings
 - Debugging: Tue:2-3pm, 5-6pm Wed: 2-3pm Thu: 9-10am, 4-5pm
 - Profiling: Tue: 3-4pm Wed: 9-10am, 4-5pm Thu: 2-3pm
- Nsight Lab
 - Tue: 4-5pm Wed: 10-11am, 3-4pm, 5-6pm Thu: 10-11am, 3-4pm
- Nsight Visual Studio Edition@NVIDIA Booth/Exhibition Hall
 - Tue, Wed: 12-2pm, 6-8pm Thu: 12-2pm