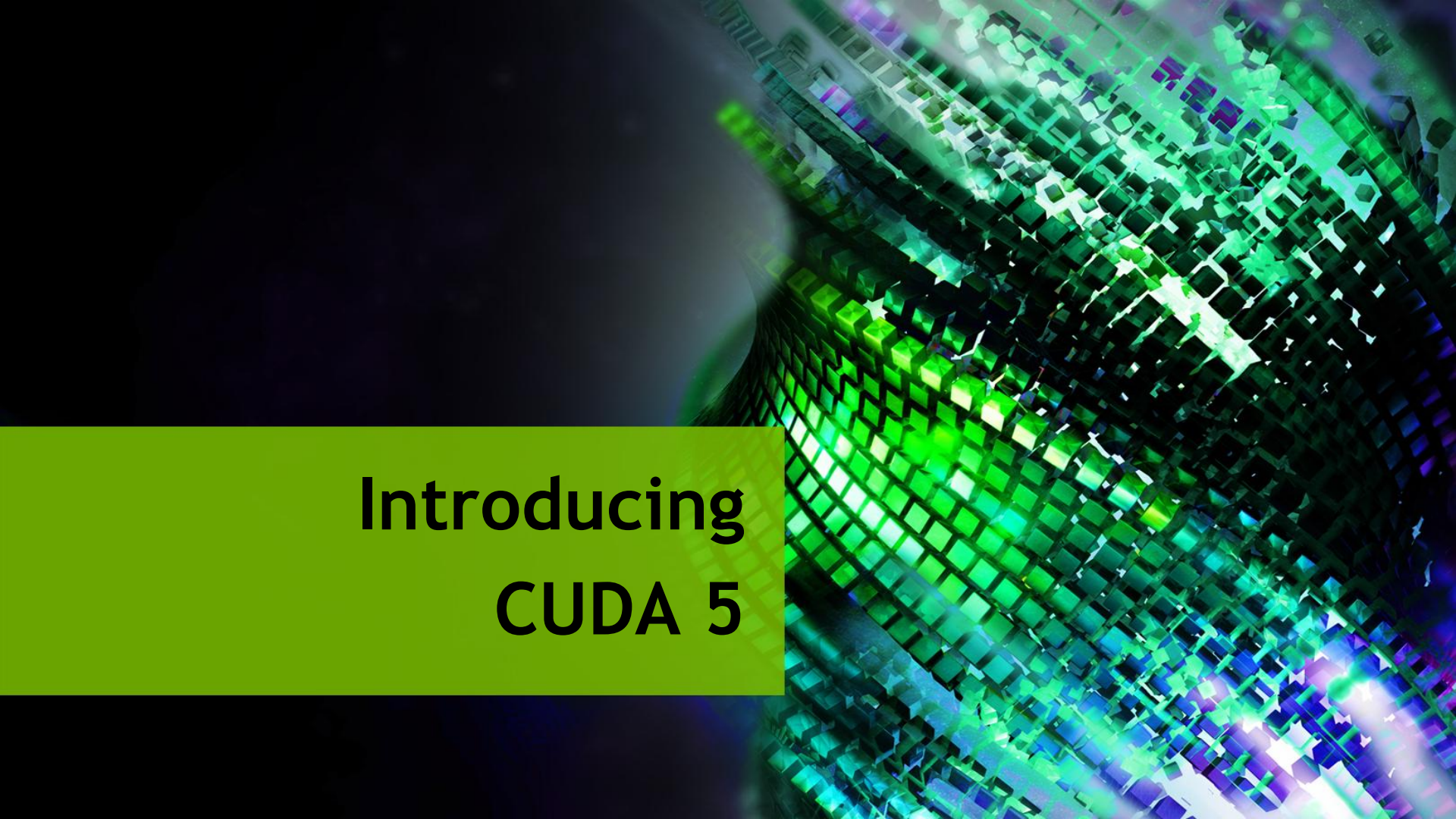

Introducing
CUDA 5

CUDA 5

Application Acceleration Made Easier

New Nsight™ Eclipse Edition

Develop, Debug, and Optimize… All in one tool!

GPUDirect™

RDMA between GPUs and PCIe devices

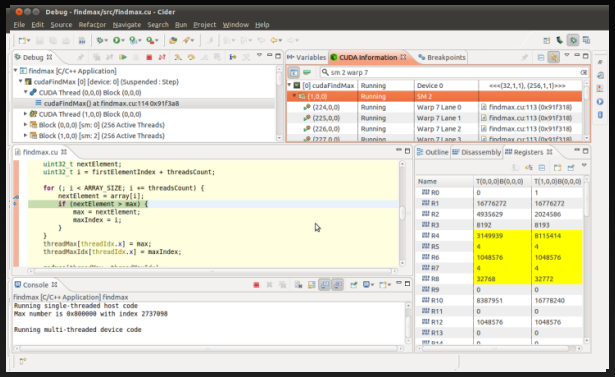
GPU Library Object Linking

Libraries and plug-ins for GPU code

Dynamic Parallelism

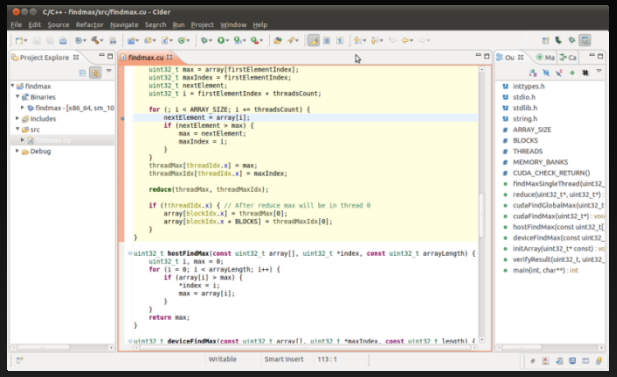
Spawn new parallel work from within GPU code on GK110

NVIDIA® Nsight,™ Eclipse Edition



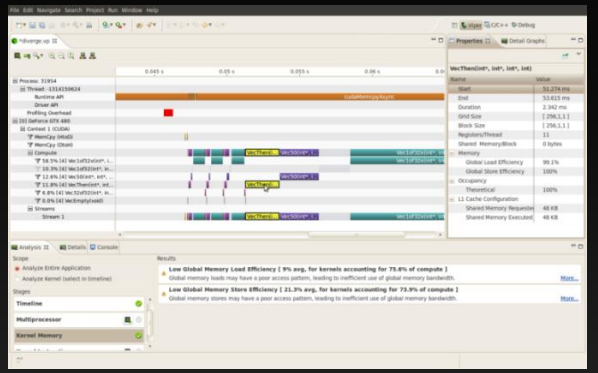
CUDA-Aware Editor

- Automated CPU to GPU code refactoring
- Semantic highlighting of CUDA code
- Integrated code samples & docs



Nsight Debugger

- Simultaneously debug of CPU and GPU
- Inspect variables across CUDA threads
- Use breakpoints & single-step debugging

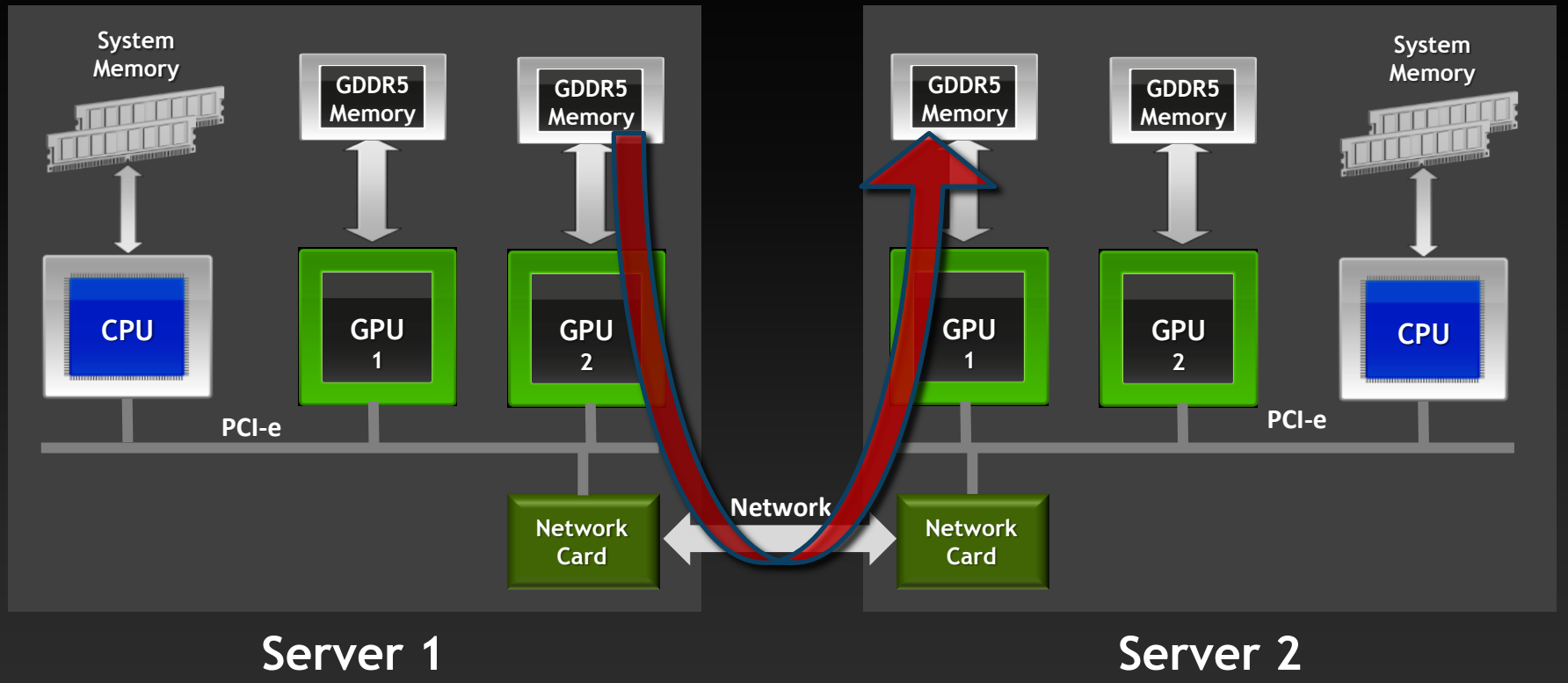


Nsight Profiler

- Quickly identifies performance issues
- Integrated expert system
- Automated analysis
- Source line correlation

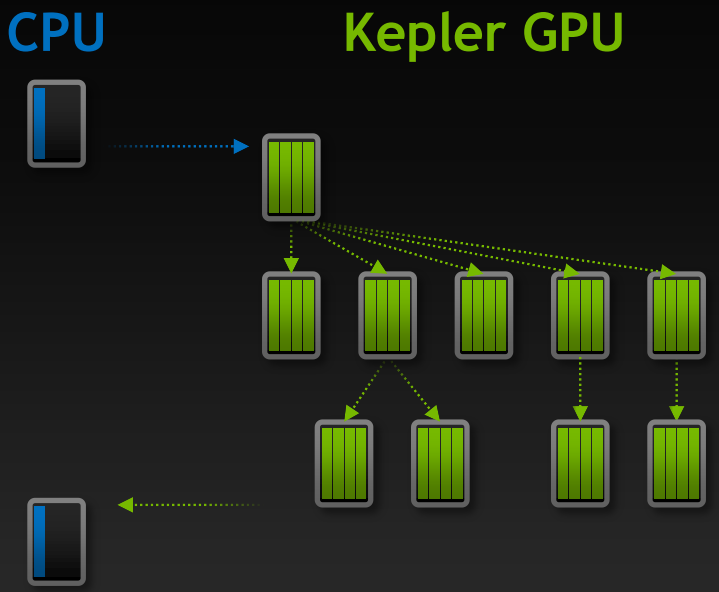
Available for Linux and Mac OS

NVIDIA GPUDirect™ Now Supports RDMA



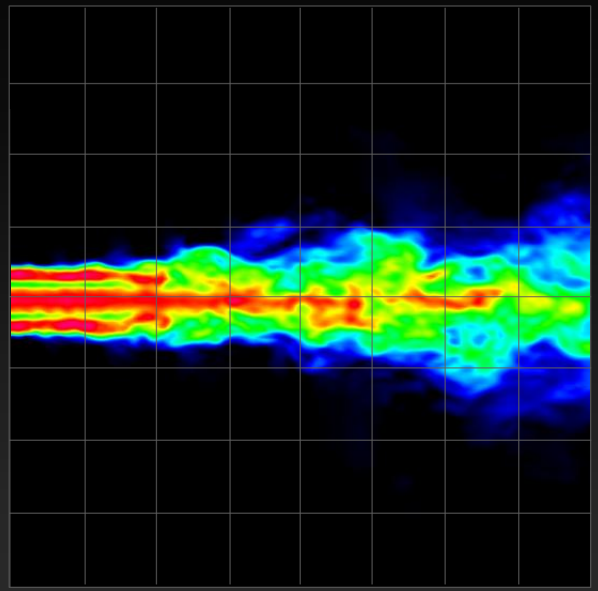
Introducing: Dynamic Parallelism

Dynamic Parallelism



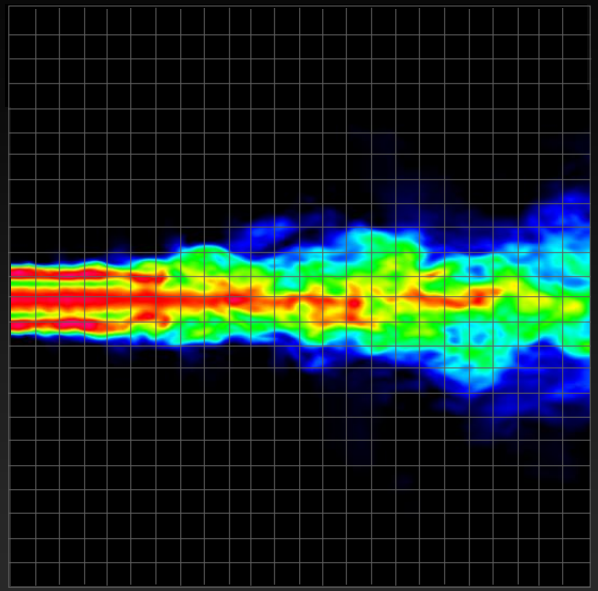
Dynamic Work Generation

Coarse grid



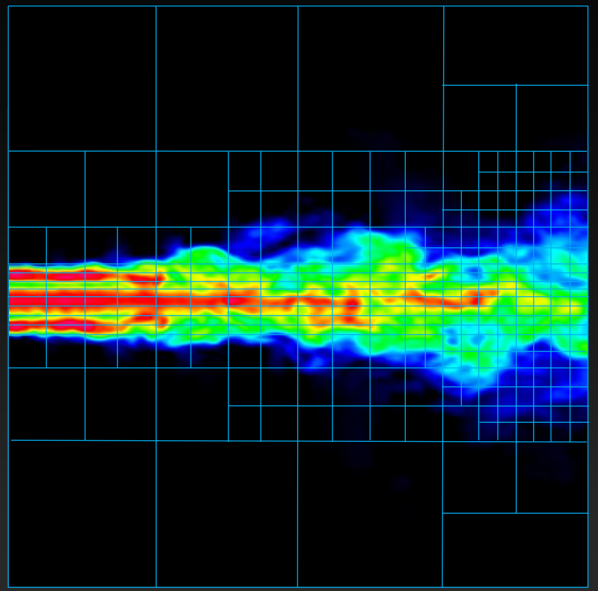
Higher Performance
Lower Accuracy

Fine grid



Lower Performance
Higher Accuracy

Dynamic grid



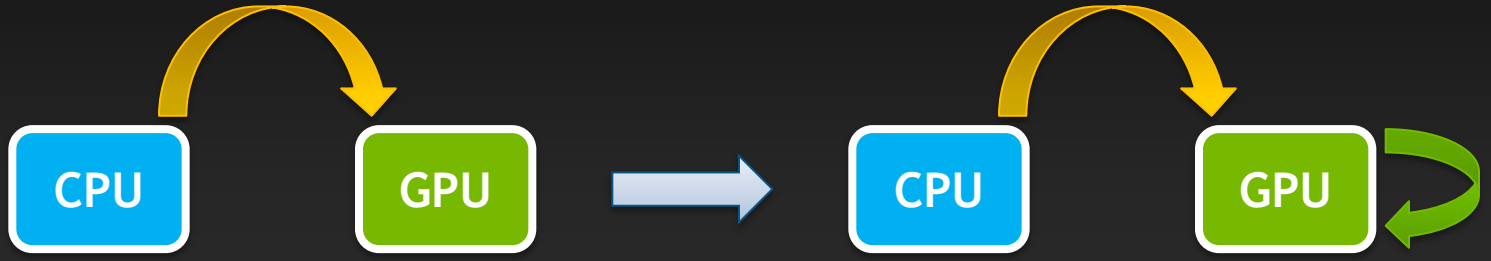
Target performance where accuracy is required

Supported on GK110 GPUs

What is Dynamic Parallelism?

The ability to launch new kernels from the GPU

- Dynamically - based on run-time data
- Simultaneously - from multiple threads at once
- Independently - each thread can launch a different grid



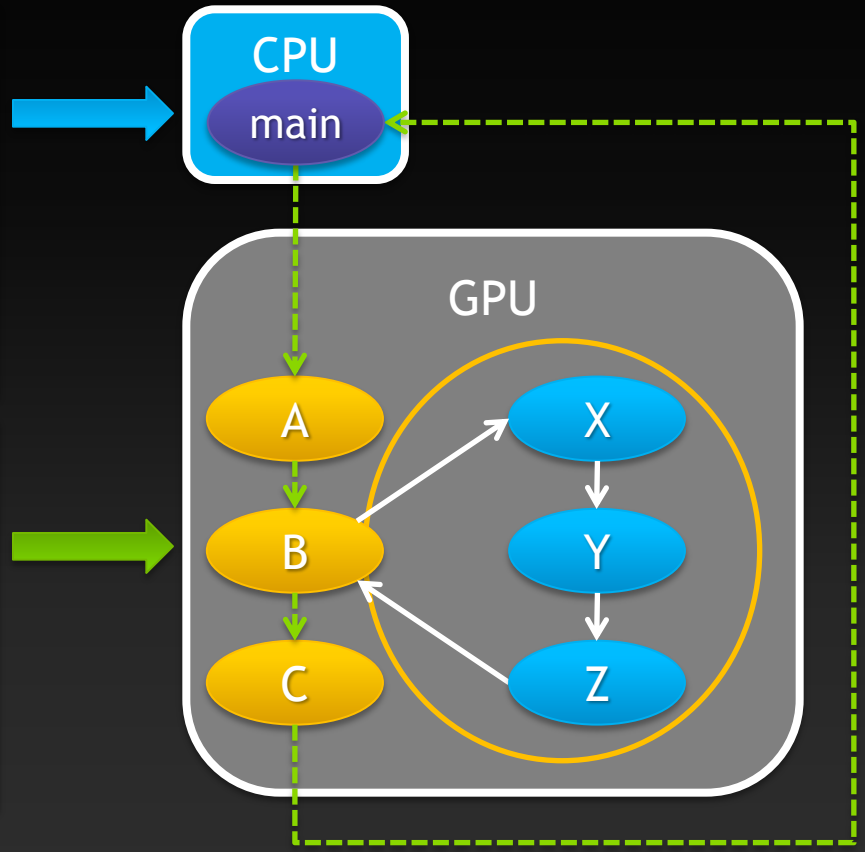
Fermi: Only CPU can generate GPU work

Kepler: GPU can generate work for itself

Familiar Syntax and Programming Model

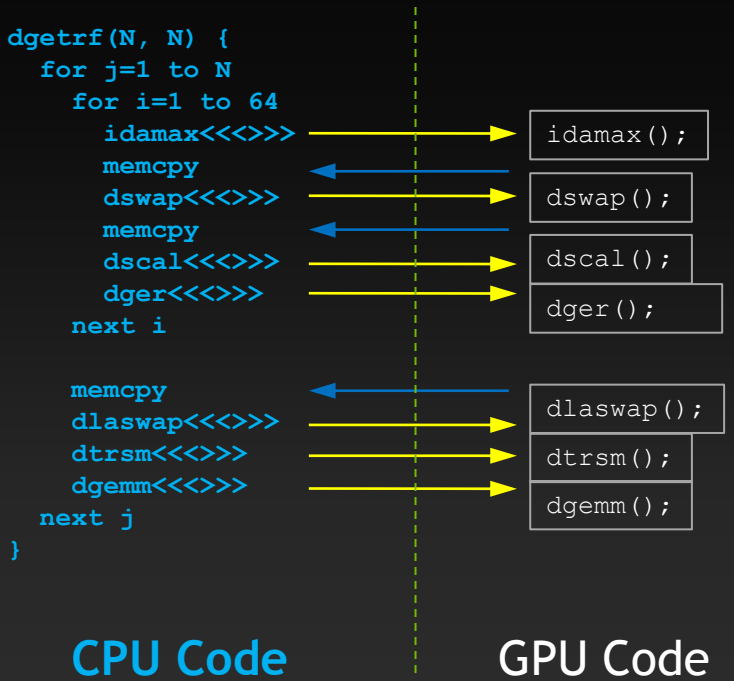
```
int main() {
    float *data;
    setup(data);

    A <<< ... >>> (data);
    B <<< ... >>> (data);
    C <<< ... >>> (data);

    cudaDeviceSynchronize();
    return 0;
}
```

```
__global__ void B(float *data)
{
    do_stuff(data);

    X <<< ... >>> (data);
    Y <<< ... >>> (data);
    Z <<< ... >>> (data);
    cudaDeviceSynchronize();

    do_more_stuff(data);
}
```

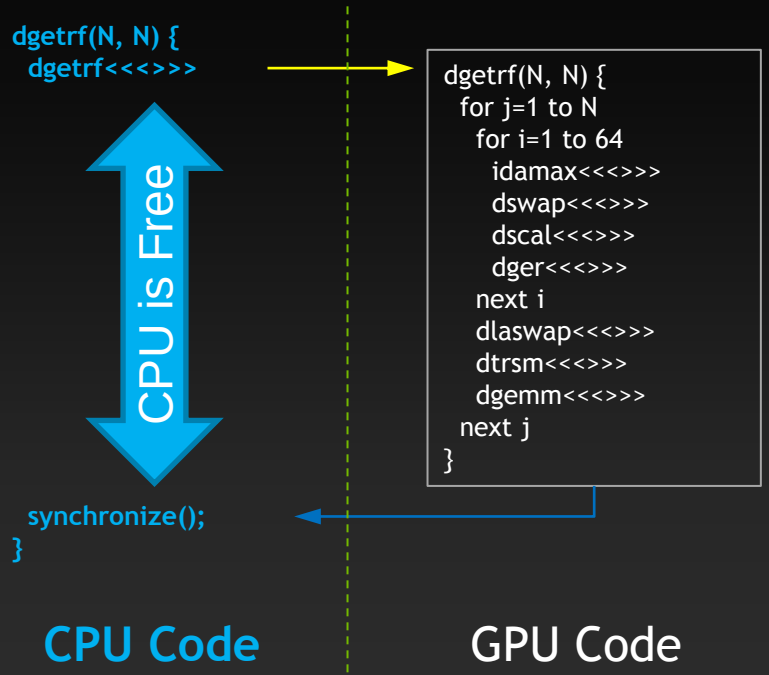


Simpler Code: LU Example

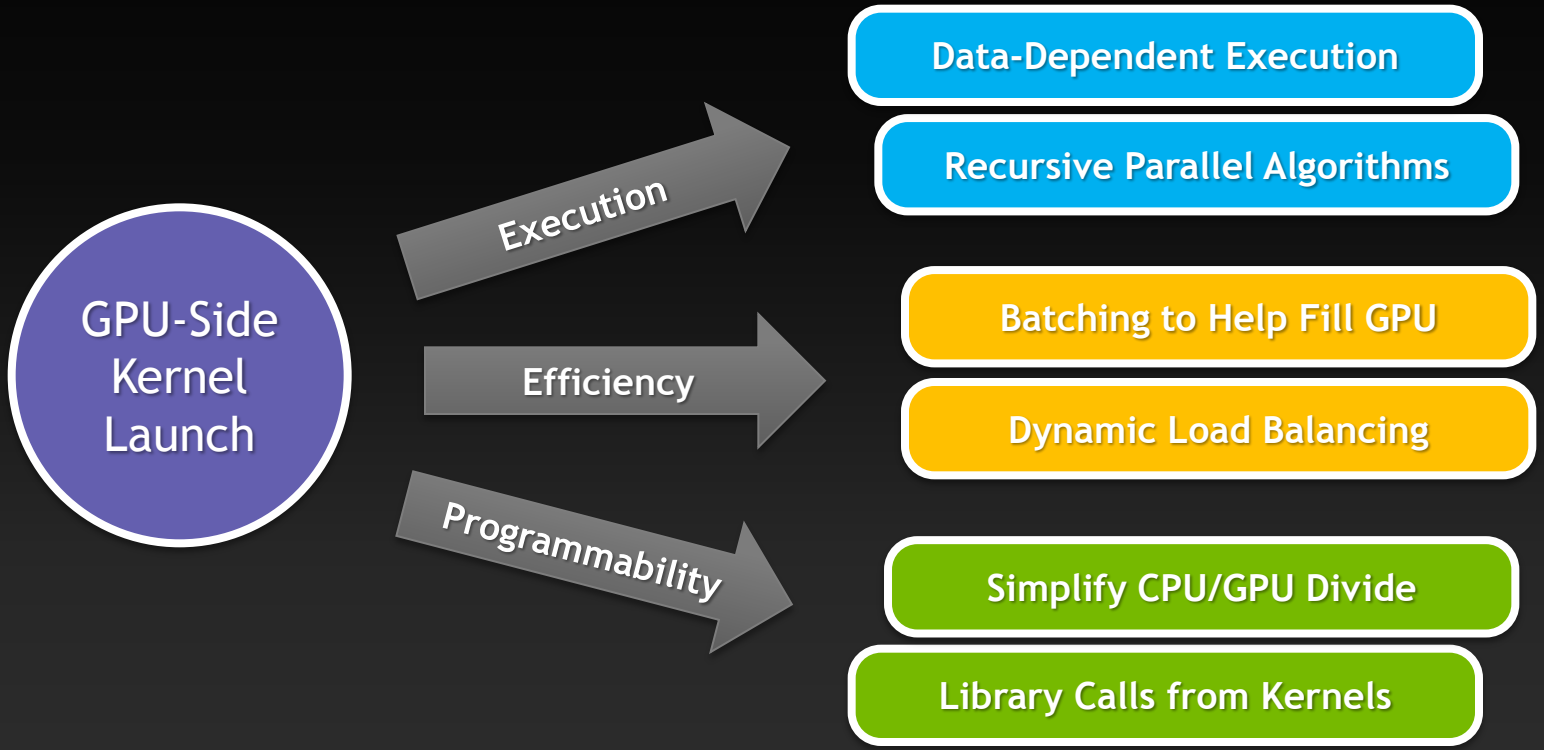
LU decomposition (Fermi)



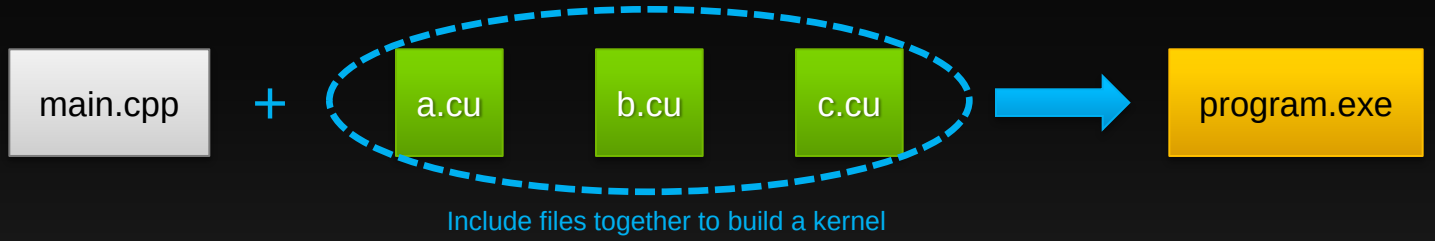
LU decomposition (Kepler)



CUDA Dynamic Parallelism



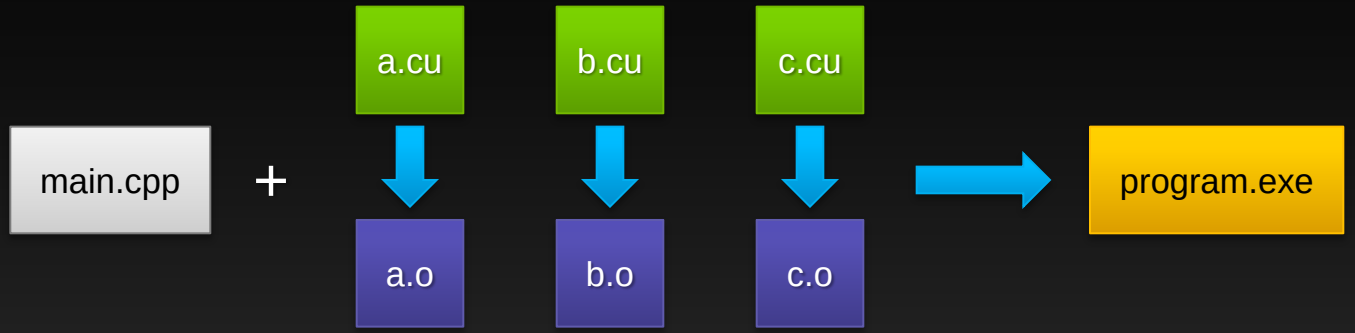
CUDA *4*: Whole-Program Compilation & Linking



CUDA 4 required all GPU kernel source included into one file

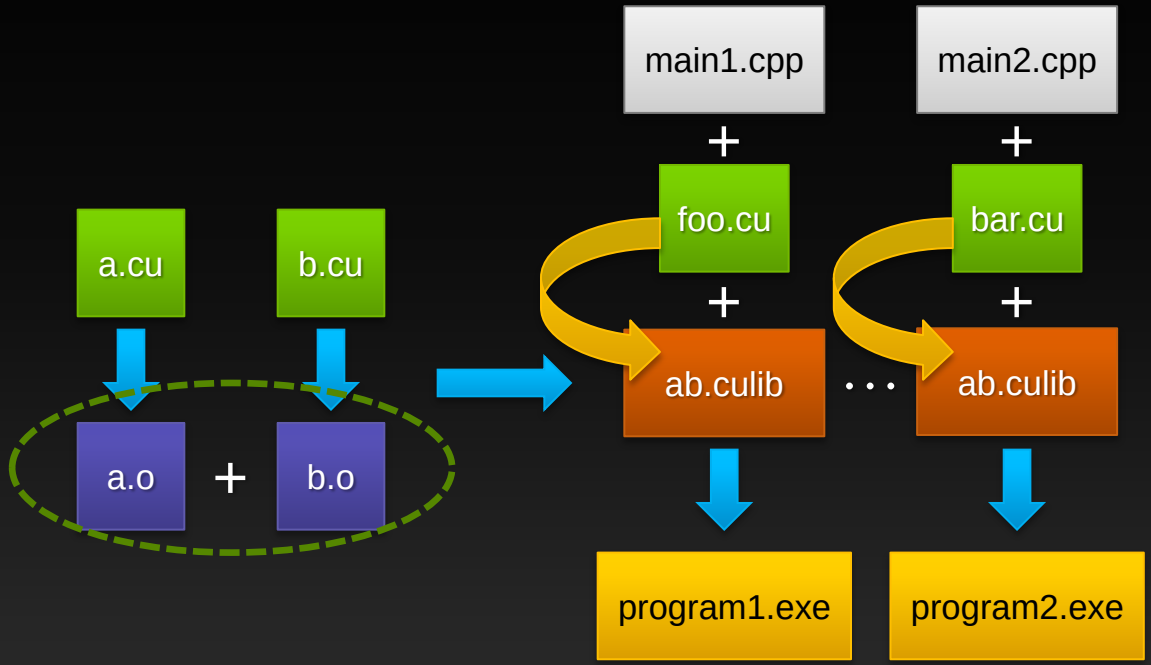
No linking external device code

CUDA 5: Separate Compilation



Separate compilation builds independent object files
Incremental compilation reduces compile time

CUDA 5: GPU Library Object Linking



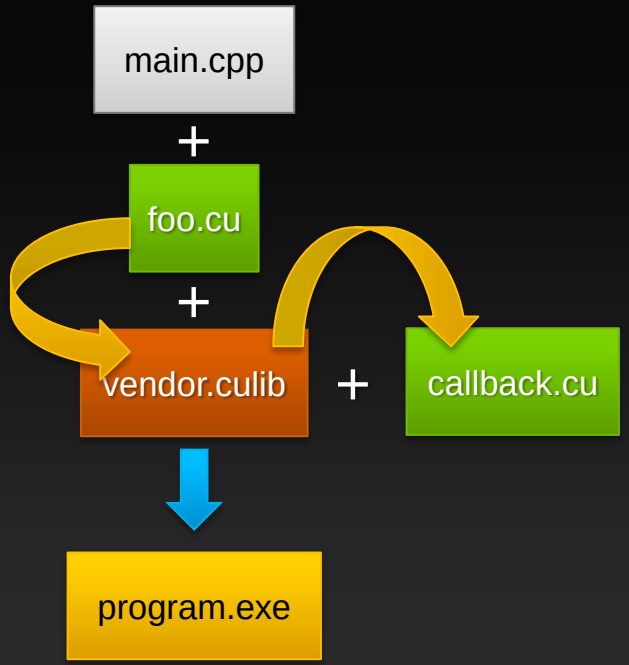
Linker combines object files into static libraries

Link and externally call *device* code

Facilitates code reuse, reduces compile time

CUDA 5: GPU Library Object Linking

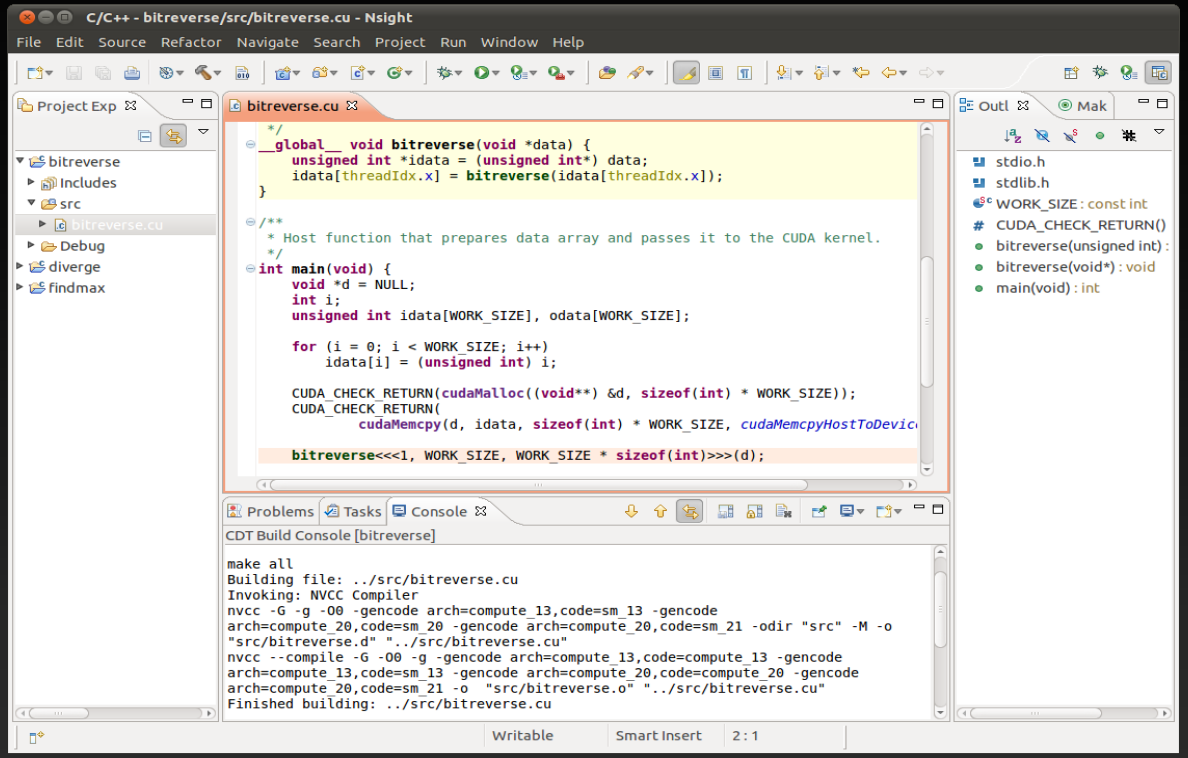
Enables 3rd party closed-source device libraries



User-defined device callback functions

Questions?

NVIDIA® Nsight,™ Eclipse Edition

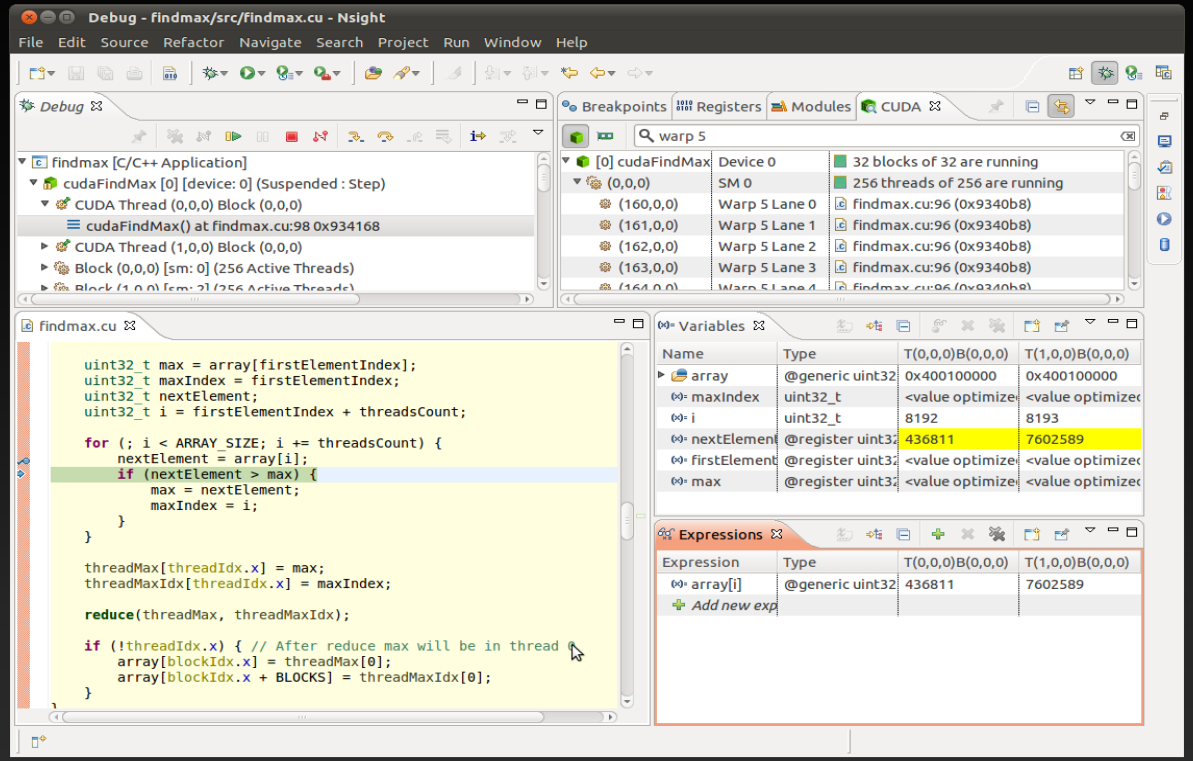


CUDA aware editor

- Integrated CUDA samples makes it quick and easy to get started
- Easily port CPU loops to CUDA kernels with automatic code refactoring
- Semantic highlighting of CUDA code makes it easy to differentiate GPU code from CPU code
- Generate code faster with CUDA aware auto code completion and inline help
- Hyperlink navigation enables faster code browsing
- Supports automatic makefile generation

www.nvidia.com/nsight

NVIDIA® Nsight,™ Eclipse Edition

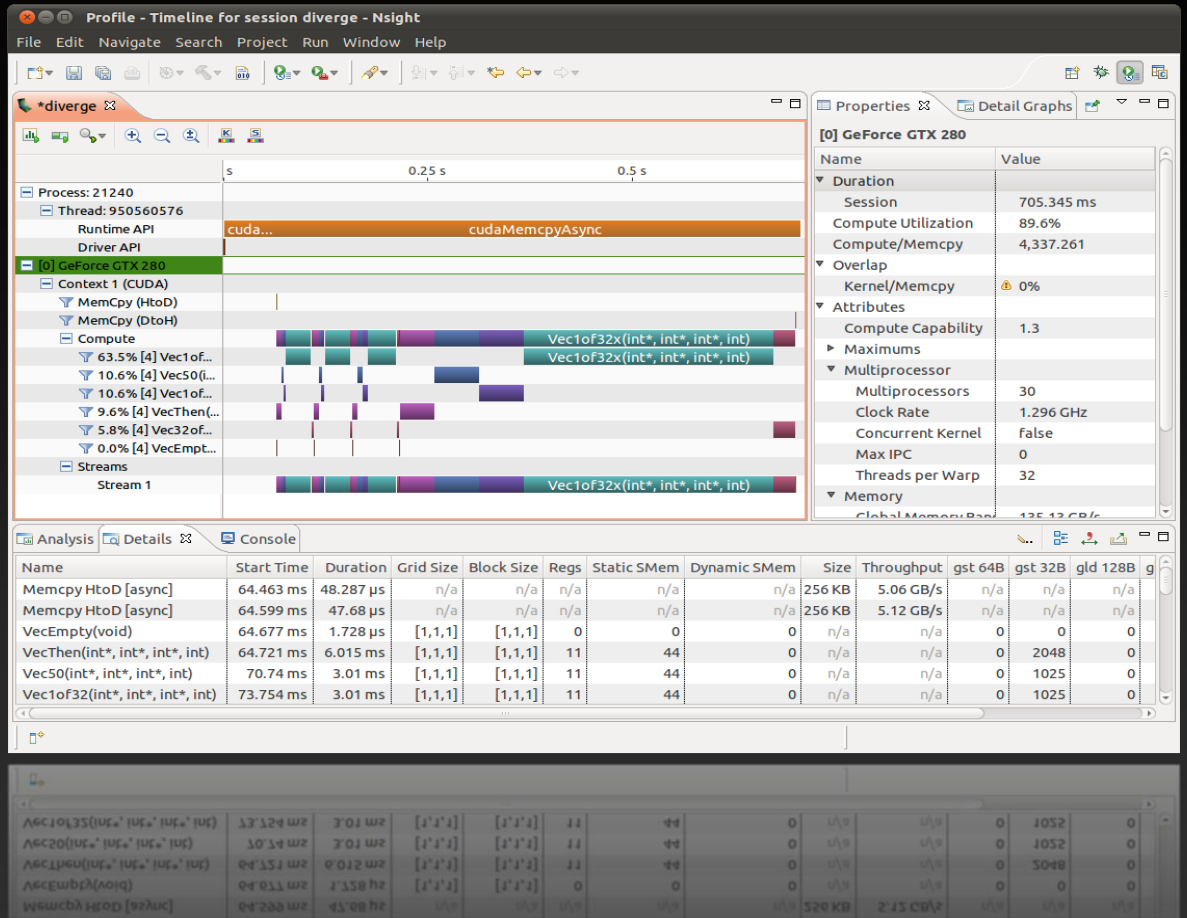


Nsight Debugger

- Seamless and simultaneous debugging of both CPU and GPU code
- View program variables across several CUDA threads
- Examine execution state and mapping of the kernels and GPUs
- View, navigate and filter to selectively track execution across threads
- Set breakpoints and single-step execution at both source-code and assembly levels
- Includes CUDA-MEMCHECK to help detect memory errors

www.nvidia.com/nsight

NVIDIA® Nsight,™ Eclipse Edition



Nsight Profiler

- Easily identify performance bottlenecks using a unified CPU and GPU trace of application activity
- Expert analysis system pin-points potential optimization opportunities
- Highlights potential performance problems at specific source-lines within application kernels
- Close integration with Nsight editor and builder for fast edit-build-profile optimization cycle
- Integrates with the new nvprof command-line profiler to enable visualization of profile data collected on headless compute nodes

www.nvidia.com/nsight

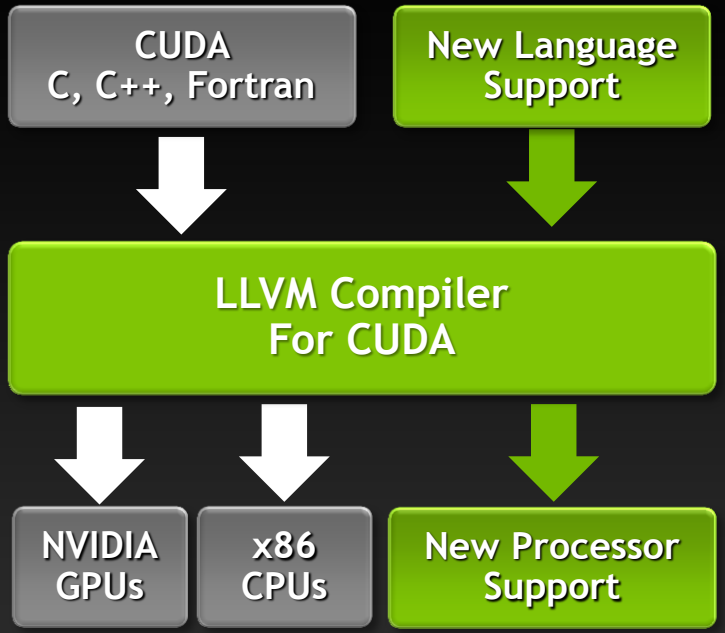
CUDA Compiler Contributed to Open Source LLVM

Developers want to build front-ends for

Java, Python, R, DSLs

Target other processors like

ARM, FPGA, GPUs, x86



LLVM COMPILER INFRASTRUCTURE

Thank you