NSIGHT DEVELOPER TOOLS
DEVELOPER TOOLS


**Profilers**: Nsight Systems, Nsight Compute, CUPTI, NVIDIA Tools eXtension (NVTX)

**Correctness Checker**: Compute Sanitizer

```bash
$ compute-sanitizer --leak-check full memcheck_demo
======== COMPUTE-SANITIZER
Mallocing memory
Running unaligned_kernel: no error
Sync: no error
Running out_of_bounds_kernel: no error
Sync: no error
Error: Invalid __global__ write of size 4 bytes
at 0x60 in memcheck_demo.cu:6:unaligned_kernel(void)
by thread (0,0,0) in block (0,0,0)
Address 0x40010000 is misaligned
```
NSIGHT PROFILING TOOLS WORKFLOW
ITERATIVE OPTIMIZATION FOR COMPUTE AND GRAPHICS

Start here

Nsight Systems
Comprehensive workload-level performance

Nsight Compute
Detailed CUDA kernel performance

Dive into top CUDA kernels by using metrics/counter collection

Nsight Graphics
Detailed frame/render performance

Dive into graphics frames

Re-check overall performance
NSIGHT SYSTEMS
SYSTEM PROFILER

Key Features:

- System-wide application algorithm tuning
  - Multi-process tree support
- Locate optimization opportunities
  - Visualize millions of events on a very fast GUI timeline
  - Or gaps of unused CPU and GPU time
- Balance your workload across multiple CPUs and GPUs
  - CPU algorithms, utilization and thread state
  - GPU streams, kernels, memory transfers, etc
- Command Line, Standalone, IDE Integration

OS: Linux (x86, Power, Arm SBSA, Tegra), Windows, MacOSX (host)
GPUs: Pascal+

Processes and threads

CUDA and OpenGL API trace

cuDNN and cuBLAS trace

Kernel and memory transfer activities

Multi-GPU

Thread/core migration

Thread state
COLLECT A PROFILE WITH NSIGHT SYSTEMS

$ nsys profile -o report --stats=true ./myapp.exe

Generated file: report.qdrep (or report.nsys-rep); open for viewing in the Nsight Systems UI

When using MPI, I recommend using nsys after mpirun/srun/jsrun/etc.:

$ mpirun -n 4 nsys profile ./myapp.exe
NSIGHT COMPUTE
KERNEL PROFILING TOOL

Key Features:
- Interactive CUDA API debugging and kernel profiling
- Built-in rules expertise
- Fully customizable data collection and display
- Command Line, Standalone, IDE Integration, Remote Targets

OS: Linux (x86, Power, Tegra, Arm SBSA), Windows, MacOSX (host only)
GPUs: Volta, Turing, Ampere GPUs

Targeted metric sections
Customizable data collection and presentation
Built-in expertise for Guided Analysis and optimization
Visual memory analysis chart

Metrics for peak performance ratios
Source/PTX/SASS analysis and correlation

Source metrics per instruction

Metric heatmap to quickly identify hotspots
KERNEL PROFILES WITH NSIGHT COMPUTE

$ ncu -k mykernel -o report ./myapp.exe

Generated file: report.ncu-rep; open for viewing in the Nsight Compute UI

(Without the -k option, Nsight Compute will profile everything and take a long time!)
PROFILING RESOURCES

Nsight Systems, Nsight Compute product pages

NVIDIA Developer Blog: NVTX

NVIDIA Developer Blog: Transitioning from nvprof to nsys

NVIDIA Developer Blog: Using Nsight Compute to Inspect Your Kernels

OLCF Training, March 2020: Nsight Compute, Nsight Systems
CUDA GDB
COMMAND LINE AND IDE BACKEND DEBUGGER

- Unified CPU and CUDA Debugging
- CUDA-C/PTX/SASS support
- Built on GDB and uses many of the same CLI commands

```plaintext
(cuda-gdb) info cuda threads breakpoint all
BlockIdx ThreadIdx Virtual PC Dev SM Wp Ln Filename Line
Kernel 0
(1,0,0) (0,0,0) 0x0000000000948e58 0 11 0 0 infoCommands.cu 12
(1,0,0) (1,0,0) 0x0000000000948e58 0 11 0 1 infoCommands.cu 12
(1,0,0) (2,0,0) 0x0000000000948e58 0 11 0 2 infoCommands.cu 12
(1,0,0) (3,0,0) 0x0000000000948e58 0 11 0 3 infoCommands.cu 12
(1,0,0) (4,0,0) 0x0000000000948e58 0 11 0 4 infoCommands.cu 12
(1,0,0) (5,0,0) 0x0000000000948e58 0 11 0 5 infoCommands.cu 12
```

```plaintext
(cuda-gdb) info cuda threads breakpoint 2 lane 1
BlockIdx ThreadIdx Virtual PC Dev SM Wp Ln Filename Line
Kernel 0
(1,0,0) (1,0,0) 0x0000000000948e58 0 11 0 1 infoCommands.cu 12
```
**COMPUTE SANITIZER**

**AUTOMATICALLY SCAN FOR BUGS AND MEMORY ISSUES**

- Compute Sanitizer checks correctness issues via sub-tools:
  - **Memcheck** - The memory access error and leak detection tool.
  - **Racecheck** - The shared memory data access hazard detection tool.
  - **Initcheck** - The uninitialized device global memory access detection tool.
  - **Synccheck** - The thread synchronization hazard detection tool.