# Profiling and Debugging

Matt Sellitto
Dana Schaa
Northeastern University
NUCAR

# AMD Performance Counters

| Counter | Description |
|---|---|
| GlobalWorkSize | The global work-item size of the kernel. |
| GroupWorkSize | The work-group size of the kernel. |
| KernelTime | Time spent executing the kernel in milliseconds (does not include the kernel setup time). |
| LocalMem | The amount of local memory in bytes being used by the kernel. |
| MemTransferSize | The memory transfer size in bytes. |
| ALU | The ALU instructions executed per thread. |
| Fetch | The fetch instructions executed per thread. |
| Write | The write instructions executed per thread. |
| Wavefront | Total number of wavefronts. |
| ALUBusy | The percentage of time ALU instructions are processed relative to GPU Time. |
| ALUFetchRatio | The ratio of ALU to Fetch instructions. |

# AMD Performance Counters

| Counter | Description |
|---|---|
| ALUPacking | The ALU vector packing efficiency (in percentage). This value indicates how well the Shader Compiler packs the scalar or vector ALU in your kernel to the five-way VLIW instructions. As a rule of thumb, values below 70 percent indicate that ALU dependency chains may be preventing full use of the processor. |
| FlowControlUtil | The kernel flow control instruction use (in percentage). As a rule of thumb, values below 70 percent suggest you could see performance improvements by reducing the number of instructions inside the flow control. |
| FetchUnitBusy | The percentage of time the Fetch unit is active relative to GPU Time. All extra fetches and any cache or memory effects are taken into account. |
| FetchUnitStalled | The percentage of time the Fetch unit is stalled relative to GPU Time. To reduce this percentage, try reducing the number of fetches or reducing the amount per fetch. |
| WriteUnitStalled | The percentage of time the Write unit is stalled relative to GPU Time. |
| **Specific to ATI Radeon™ HD 5000 Series Graphics Cards** | |
| ALUStalledByLDS | The percentage of GPU Time ALU units are stalled because the LDS input queue is full and the output queue is not ready. To reduce this percentage, try to minimize the number of LDS bank conflicts, or reduce the total number of LDS accesses. |
| LDSBankConflict | The percentage of GPU Time the LDS is stalled by bank conflicts. |

# AMD Performance Counters

- *FetchMem* – data read from global memory
- *FastPath* and *CompletePath* – data written to global memory (two different buses)
  - FastPath is an optimized hardware path, but supports only simple operations (32-bit+ data types, no atomics)
    - 100+ GB/s measured throughput
  - CompletePath handles sub-32-bit data and advanced operations
    - 20 GB/s measured throughput
  - This bus design is specific to AMD

# AMD Performance Counters

- *L1CacheHit* – percentage of fetches that hit in L1 memory
  - Texture memory is cached on AMD systems

# Performance Counters

- Running Example: Matrix Addition
- Data set information
  - 2048 x 2048 threads created
  - 64 threads/wavefront
  - 65536 wavefronts

| Method | GlobalWorkSize | GroupWorkSize | Wavefront |
|---|---|---|---|
| matrixadd_086D5C08 | { 2048 2048 1} | { 1 64 1} | 65536 |
| matrixadd_086D5C08 | { 2048 2048 1} | { 64 1 1} | 65536 |
| matrixadd_086D5C08 | { 2048 2048 1} | { 8 8 1} | 65536 |

# Performance Counters

- Fetch Mem (data read from global memory)
  - For AMD, memory transactions are based on quarter-wavefronts (currently)
    - 64-byte reads
  - The workgroup dimensions affect the amount of useful data read each access
    - Reading down a column creates bank conflicts
    - For worst case 4 useful bytes per read (or 1/16th of memory transaction)
    - For 8x8 case, ½ of data is useful

| Method | GroupWorkSize | FetchMem |
|---|---|---|
| matrixadd_086D5C08 | { 1 64 1} | 524288 |
| matrixadd_086D5C08 | { 64 1 1} | 32768 |
| matrixadd_086D5C08 | { 8 8 1} | 65536 |

# Performance Counters

- Memory writes
  - No as much information is available about bus design
  - No extra penalty for writing 8x8 workgroup
  - Just over 4X penalty in bytes written for worst-case scenario

| Method | GroupWorkSize | FastPath | CompletePath |
|---|---|---|---|
| matrixadd_086D5C08 | { 1 64 1} | 71822.88 | 0 |
| matrixadd_086D5C08 | { 64 1 1} | 16383 | 0 |
| matrixadd_086D5C08 | { 8 8 1} | 16383.50 | 0 |

# Performance Counters

- Memory access pattern has a huge effect on performance
  - Biggest performance bottleneck if done incorrectly
- Memory reads from quarter-wavefronts (16 threads) can be coalesced
  - Single memory address sent, multiple data items are retrieved

| Method | GroupWorkSize | FetchMem | FastPath | CompletePath | Time |
|---|---|---|---|---|---|
| matrixadd_086D5C08 | { 1 64 1} | 524288 | 71822.88 | 0 | 73.89982 |
| matrixadd_086D5C08 | { 64 1 1} | 32768 | 16383 | 0 | 0.42722 |
| matrixadd_086D5C08 | { 8 8 1} | 65536 | 16383.50 | 0 | 1.50243 |

**176X faster**

# Performance Counters

- SIMD utilization (i.e., useful work)
  - Amount of time ALUs are busy (ALUBusy) * how full they are (ALUPacking)
  - Low utilization implies memory latency or not enough threads scheduled

| ALUBusy | ALUPacking |
|---|---|
| 0.19 | 35.56 |
| 32.45 | 35.56 |
| 9.34 | 35.56 |

# Memory Bandwidth

- Effective memory BW = (Br + Bw)/T
- Br = total number of bytes read
- Bw = total number of bytes written
- T = time required to run the kernel

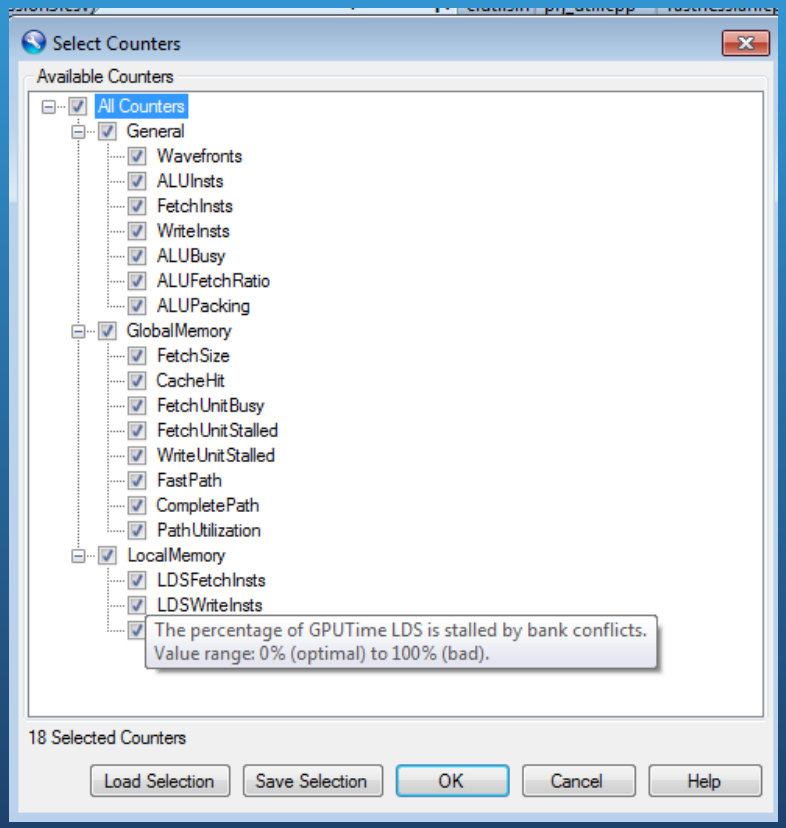
| WG Dims | 1x64 | 8x8 | 64x1 |
|---|---|---|---|
| Br | 512MB | 64MB | 32MB |
| Bw | 70.1MB | 16MB | 16MB |
| T | 73.9ms | 1.5ms | 0.42ms |
| $BW_{eff}$ | 7.7GB/s | 52.1GB/s | 111.6GB/s |

# AMD Stream Profiler

- Dynamic profiler (GPU and CPU)
- Visual Studio plug-in
- Reports performance counters from devices and static information from the binary
  - Works in a limited scope for the CPU (mostly just static information)

# AMD Stream Profiler

- Selecting performance counters



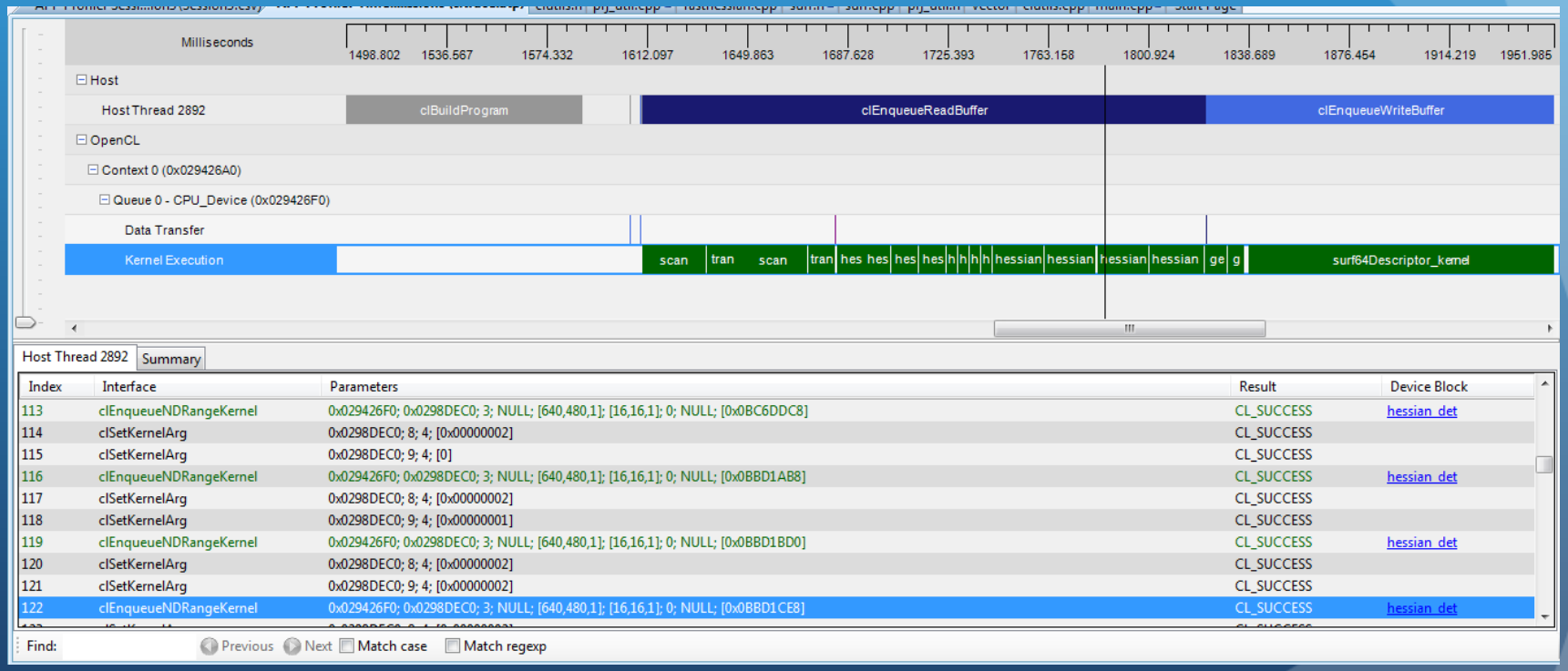
# AMD Stream Profiler

- Performance counters

| Method | ExecutionOrder | GlobalWorkSize | GroupWorkSize | Time | LocalMemSize | DataTransferSize | GPRs | ScratchRegs | FCStacks | Wavefronts | ALUInsts |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CreateBuffer | 7 | | | NA | | 0.06 | | | | | |
| CreateBuffer | 8 | | | NA | | 0.06 | | | | | |
| WriteBuffer | 9 | | | 0.56145 | | 1200 | | | | | |
| scan4__k1_Cayman1 | 10 | { 64 480 1} | { 64 1 1} | 0.10590 | 3072 | | 8 | 0 | 3 | 480 | 308 |
| transpose__k2_Cayman1 | 11 | { 640 480 1} | { 16 16 1} | 0.09592 | 1024 | | 3 | 0 | 2 | 4800 | 31 |
| scan4__k1_Cayman1 | 12 | { 64 640 1} | { 64 1 1} | 0.05669 | 3072 | | 8 | 0 | 3 | 640 | 213 |
| transpose__k2_Cayman1 | 13 | { 480 640 1} | { 16 16 1} | 0.03982 | 1024 | | 3 | 0 | 2 | 4800 | 31 |
| CopyBufferToImage2D | 14 | | | 0.21064 | | 1200 | | | | | |
| hessian_det__k3_Cayman1 | 15 | { 640 480 1} | { 16 16 1} | 1.22999 | 0 | | 10 | 0 | 2 | 4800 | 63.24 |
| hessian_det__k3_Cayman1 | 16 | { 640 480 1} | { 16 16 1} | 0.09283 | 0 | | 10 | 0 | 2 | 4800 | 63.24 |
| hessian_det__k3_Cayman1 | 17 | { 640 480 1} | { 16 16 1} | 0.09391 | 0 | | 10 | 0 | 2 | 4800 | 63.24 |
| hessian_det__k3_Cayman1 | 18 | { 640 480 1} | { 16 16 1} | 0.09867 | 0 | | 10 | 0 | 2 | 4800 | 63.24 |
| hessian_det__k3_Cayman1 | 19 | { 640 480 1} | { 16 16 1} | 0.10203 | 0 | | 10 | 0 | 2 | 4800 | 48.89 |
| hessian_det__k3_Cayman1 | 20 | { 640 480 1} | { 16 16 1} | 0.10095 | 0 | | 10 | 0 | 2 | 4800 | 48.89 |
| hessian_det__k3_Cayman1 | 21 | { 640 480 1} | { 16 16 1} | 0.10777 | 0 | | 10 | 0 | 2 | 4800 | 48.89 |
| hessian_det__k3_Cayman1 | 22 | { 640 480 1} | { 16 16 1} | 0.10993 | 0 | | 10 | 0 | 2 | 4800 | 48.89 |
| hessian_det__k3_Cayman1 | 23 | { 640 480 1} | { 16 16 1} | 0.07941 | 0 | | 10 | 0 | 2 | 4800 | 42.24 |
| hessian_det__k3_Cayman1 | 24 | { 640 480 1} | { 16 16 1} | 0.06902 | 0 | | 10 | 0 | 2 | 4800 | 42.24 |
| hessian_det__k3_Cayman1 | 25 | { 640 480 1} | { 16 16 1} | 0.08277 | 0 | | 10 | 0 | 2 | 4800 | 42.24 |
| hessian_det__k3_Cayman1 | 26 | { 640 480 1} | { 16 16 1} | 0.07151 | 0 | | 10 | 0 | 2 | 4800 | 42.24 |

# AMD Stream Profiler

- Program trace




| Index | Interface | Parameters | Result | Device Block |
|---|---|---|---|---|
| 113 | clEnqueueNDRangeKernel | 0x029426F0; 0x0298DEC0; 3; NULL; [640,480,1]; [16,16,1]; 0; NULL; [0x0BC6DDC8] | CL_SUCCESS | hessian_det |
| 114 | clSetKernelArg | 0x0298DEC0; 8; 4; [0x00000002] | CL_SUCCESS | |
| 115 | clSetKernelArg | 0x0298DEC0; 9; 4; [0] | CL_SUCCESS | |
| 116 | clEnqueueNDRangeKernel | 0x029426F0; 0x0298DEC0; 3; NULL; [640,480,1]; [16,16,1]; 0; NULL; [0x0BBD1AB8] | CL_SUCCESS | hessian_det |
| 117 | clSetKernelArg | 0x0298DEC0; 8; 4; [0x00000002] | CL_SUCCESS | |
| 118 | clSetKernelArg | 0x0298DEC0; 9; 4; [0x00000001] | CL_SUCCESS | |
| 119 | clEnqueueNDRangeKernel | 0x029426F0; 0x0298DEC0; 3; NULL; [640,480,1]; [16,16,1]; 0; NULL; [0x0BBD1BD0] | CL_SUCCESS | hessian_det |
| 120 | clSetKernelArg | 0x0298DEC0; 8; 4; [0x00000002] | CL_SUCCESS | |
| 121 | clSetKernelArg | 0x0298DEC0; 9; 4; [0x00000002] | CL_SUCCESS | |
| 122 | clEnqueueNDRangeKernel | 0x029426F0; 0x0298DEC0; 3; NULL; [640,480,1]; [16,16,1]; 0; NULL; [0x0BBD1CE8] | CL_SUCCESS | hessian_det |

# NVIDIA Visual Profiler

- Similar to AMD Stream Profiler
  - View timing information
  - Presents hardware utilization statistics (occupancy)
    - Register usage, % of max threads, etc
  - Provides performance counters
- PROTIP: If you don’t release all of your OpenCL resources, the profiler throws a cryptic error message

# NVIDIA Visual Profiler

CUDAF - CUDA Visual Profiler - [Test1 - Device_0 - Context_0] (on danger3)

File Session View Options Window Help

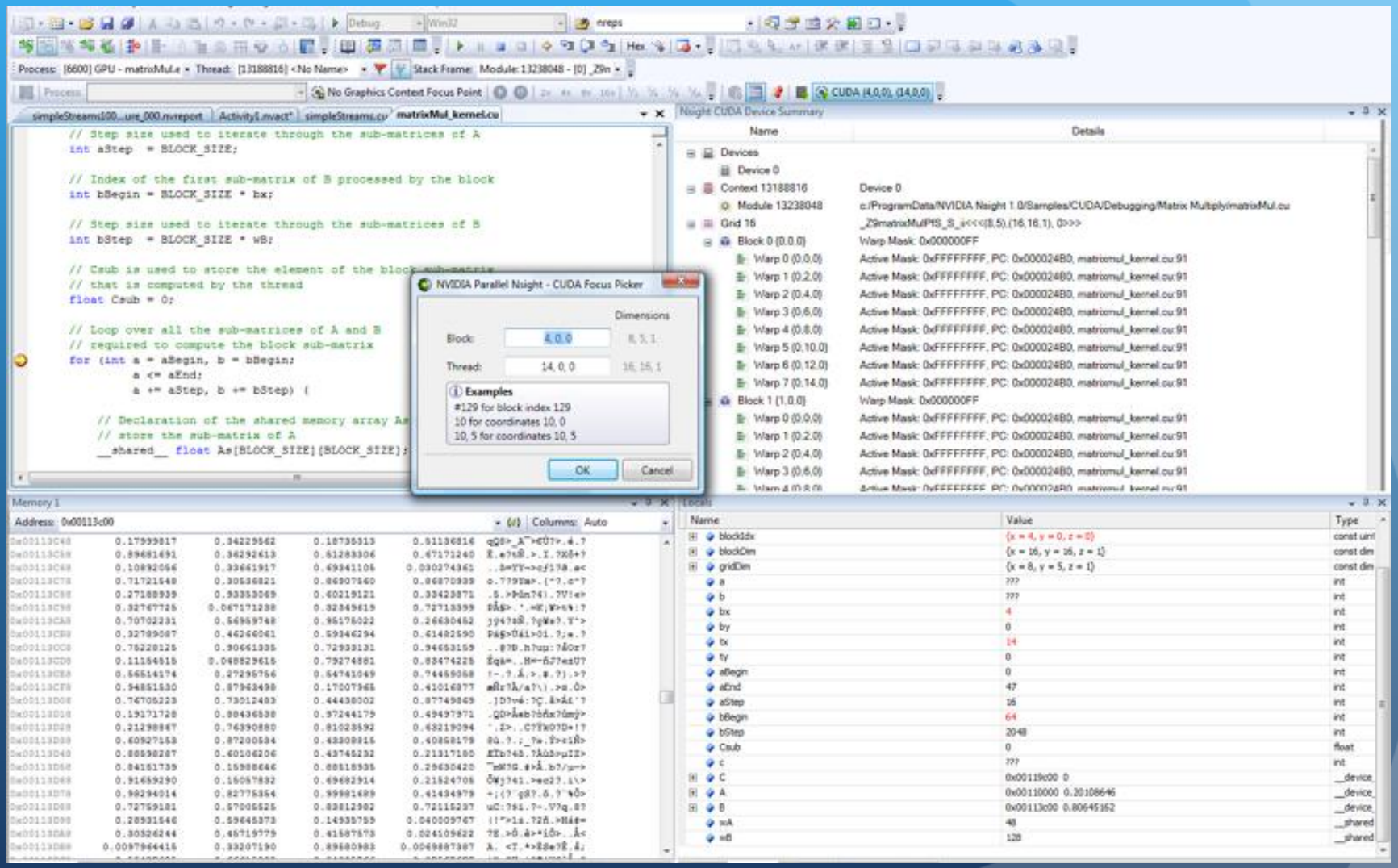
Profiler Output

| | GPU Timestamp usec | Method | GPU Time | CPU Time | Occupancy | grid size X | block size X | static shared memory per block | registers per thre |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | memcpyHtoD | 7002.34 | 7186 | | | | | |
| 2 | 13694.7 | memcpyHtoD | 6994.08 | 7177 | | | | | |
| 3 | 20790.3 | compute1_kernel | 2069.57 | 2090 | 1 | 39063 | 256 | 44 | 9 |
| 4 | 22909.7 | memcpyDtoH | 7387.52 | 7613 | | | | | |
| 5 | 45311.2 | memcpyHtoD | 6994.46 | 7177 | | | | | |
| 6 | 58975.2 | memcpyHtoD | 6987.26 | 7171 | | | | | |
| 7 | 66030.6 | compute1_kernel | 2068.96 | 2082 | 1 | 39063 | 256 | 44 | 9 |
| 8 | 68146.2 | memcpyDtoH | 7391.36 | 7612 | | | | | |
| 9 | 90497 | memcpyHtoD | 6998.08 | 7183 | | | | | |
| 10 | 104164 | memcpyHtoD | 7010.66 | 7194 | | | | | |
| 11 | 111241 | compute1_kernel | 2068.38 | 2082 | 1 | 39063 | 256 | 44 | 9 |
| 12 | 113357 | memcpyDtoH | 7396.1 | 7620 | | | | | |
| 13 | 135712 | memcpyHtoD | 7209.31 | 7391 | | | | | |
| 14 | 149602 | memcpyHtoD | 6995.46 | 7181 | | | | | |

# NVIDIA Parallel NSIGHT Debugger

- Parallel NSIGHT Debugger
  - Live GPU Debugger
    - Not available for OpenCL
  - Can set breakpoints in GPU code, single step through code
  - Can view memory contents (global, shared, local memory)

# NVIDIA Parallel NSIGHT Debugger



# NVIDIA Parallel NSIGHT Analyzer

- Visualization of program execution flow
  - Kernel calls
  - Driver functions calls
  - Memory transfers
- Similar to AMD Stream Profiler

# NVIDIA Parallel NSIGHT Analyzer