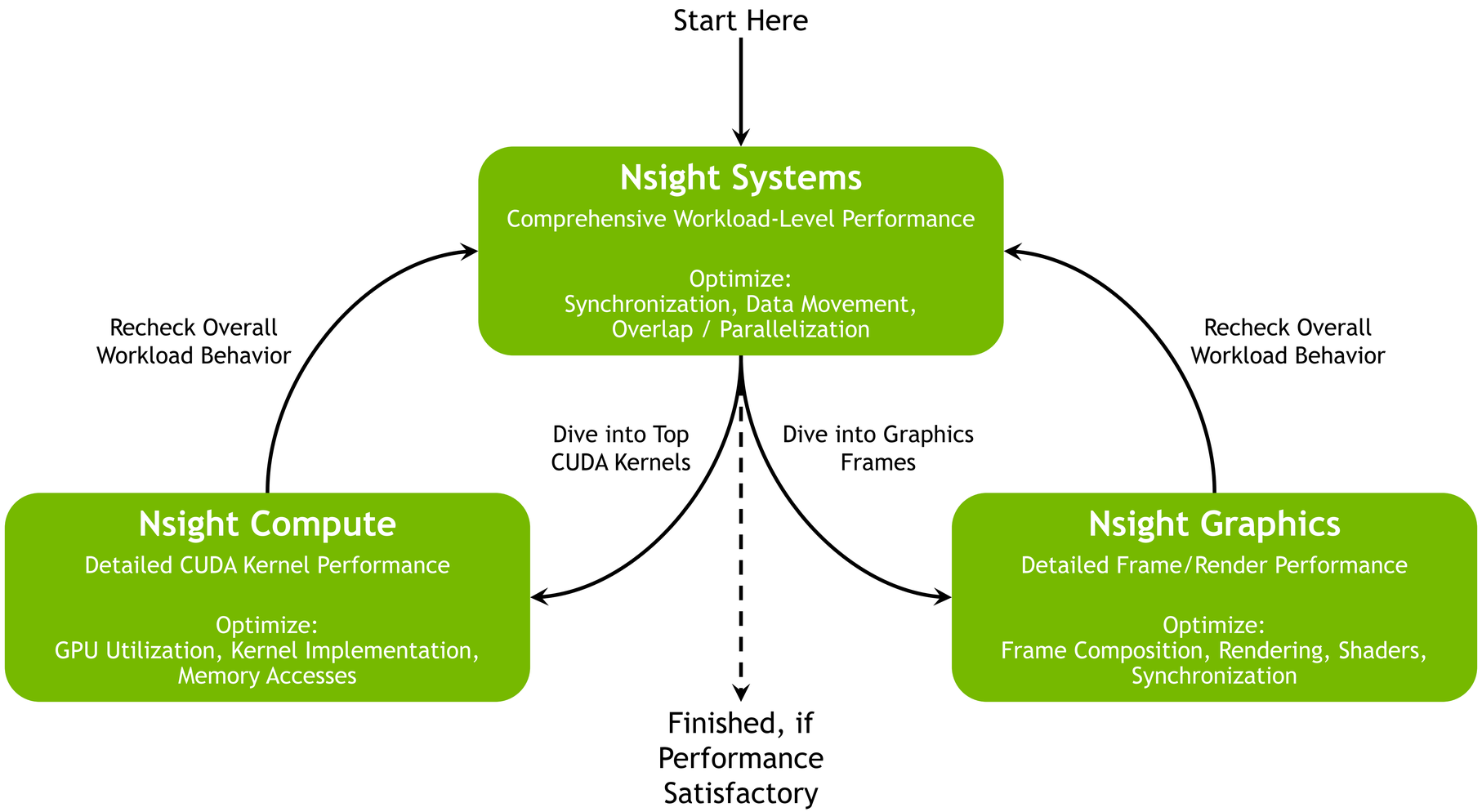
S41723

HOW TO UNDERSTAND AND OPTIMIZE SHARED MEMORY ACCESSES USING NSIGHT COMPUTE

MAGNUS STRENGERT

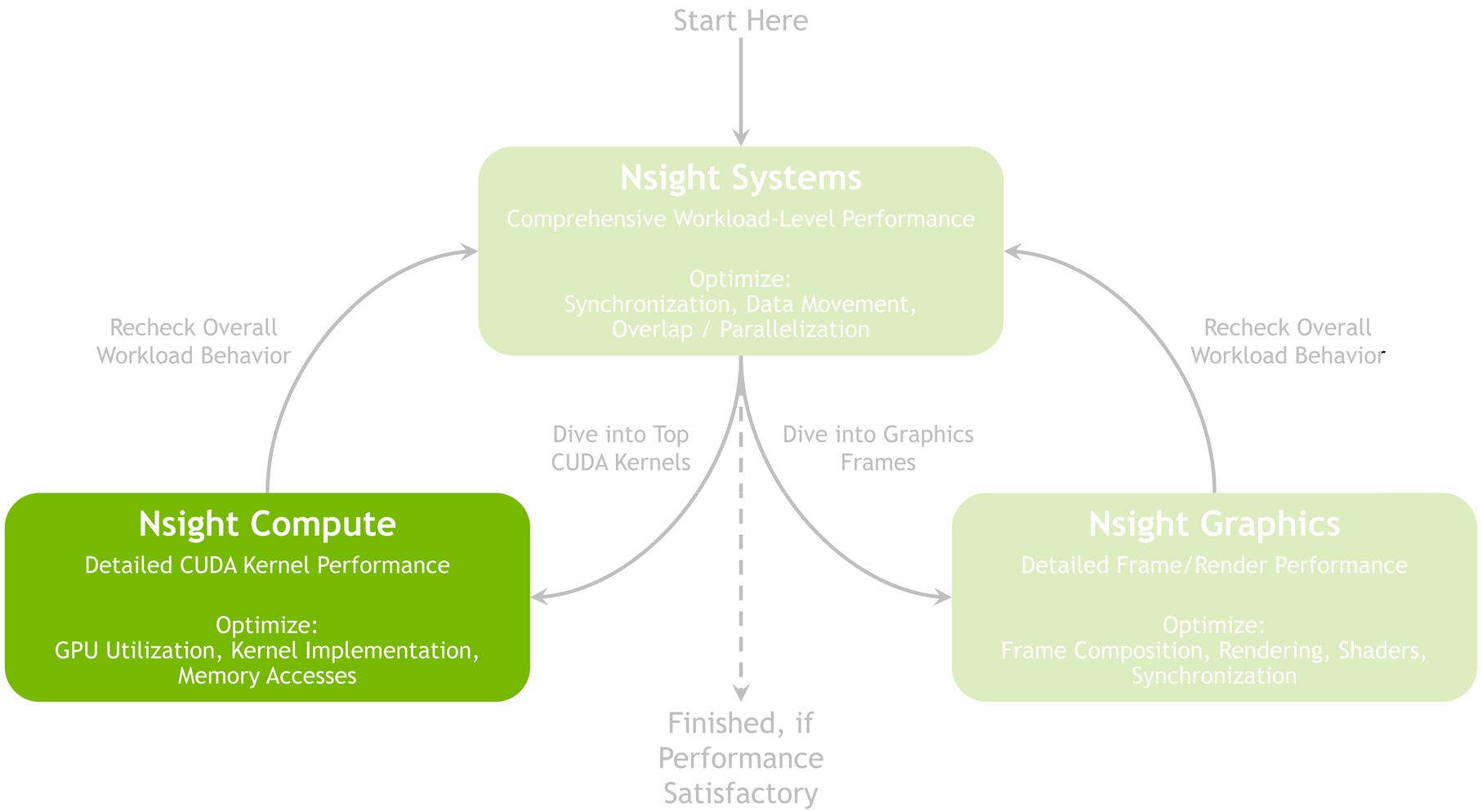
NSIGHT TOOLS

Overview



NSIGHT TOOLS

Overview



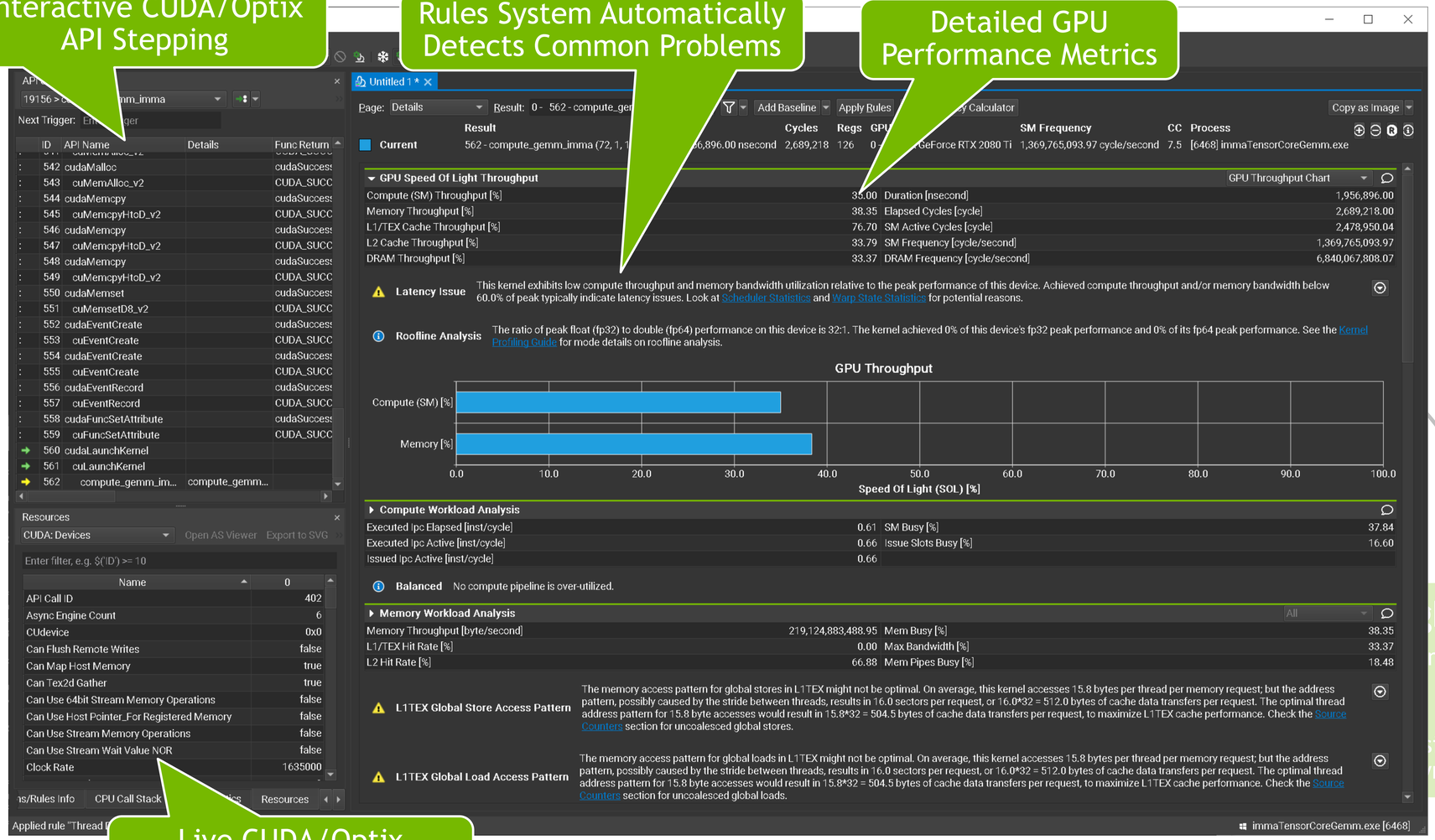
NSIGHT TOOLS

Overview

Interactive CUDA/Optix API Stepping

Rules System Automatically Detects Common Problems

Detailed GPU Performance Metrics



Live CUDA/Optix Resource Tracking

Recheck Overall Workload Behavior

ht Graphics

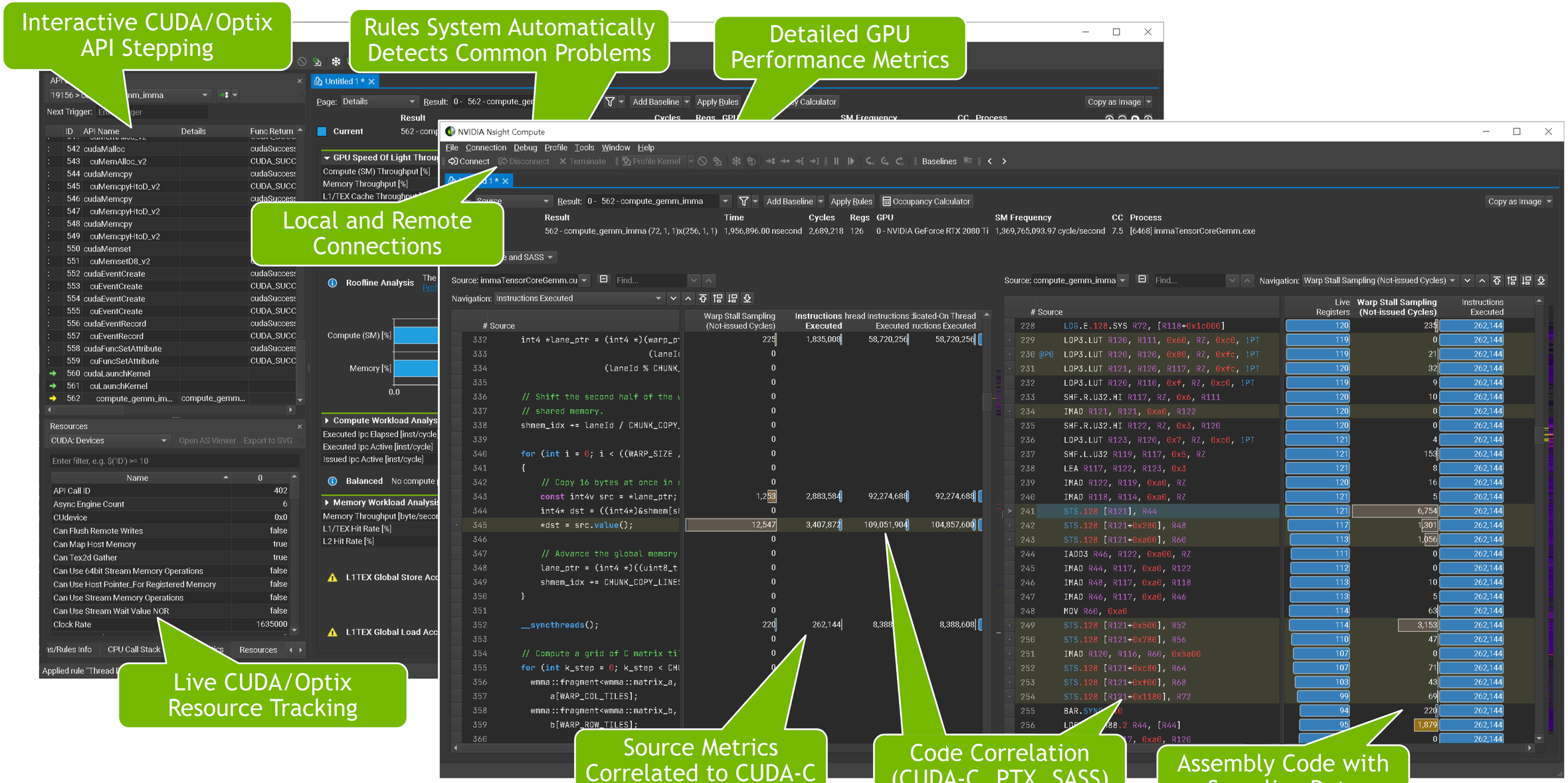
me/Render Performance

Optimize:
sition, Rendering, Shaders,
nchronization

Performance Satisfactory

NSIGHT TOOLS

Overview



NSIGHT TOOLS

Overview

Interactive CUDA/Optix API Stepping

Rules System Automatically Detects Common Problems

Detailed GPU Performance Metrics

Local and Remote Connections

Live CUDA/Optix Resource Tracking

Source Metrics Correlated to CUDA-C

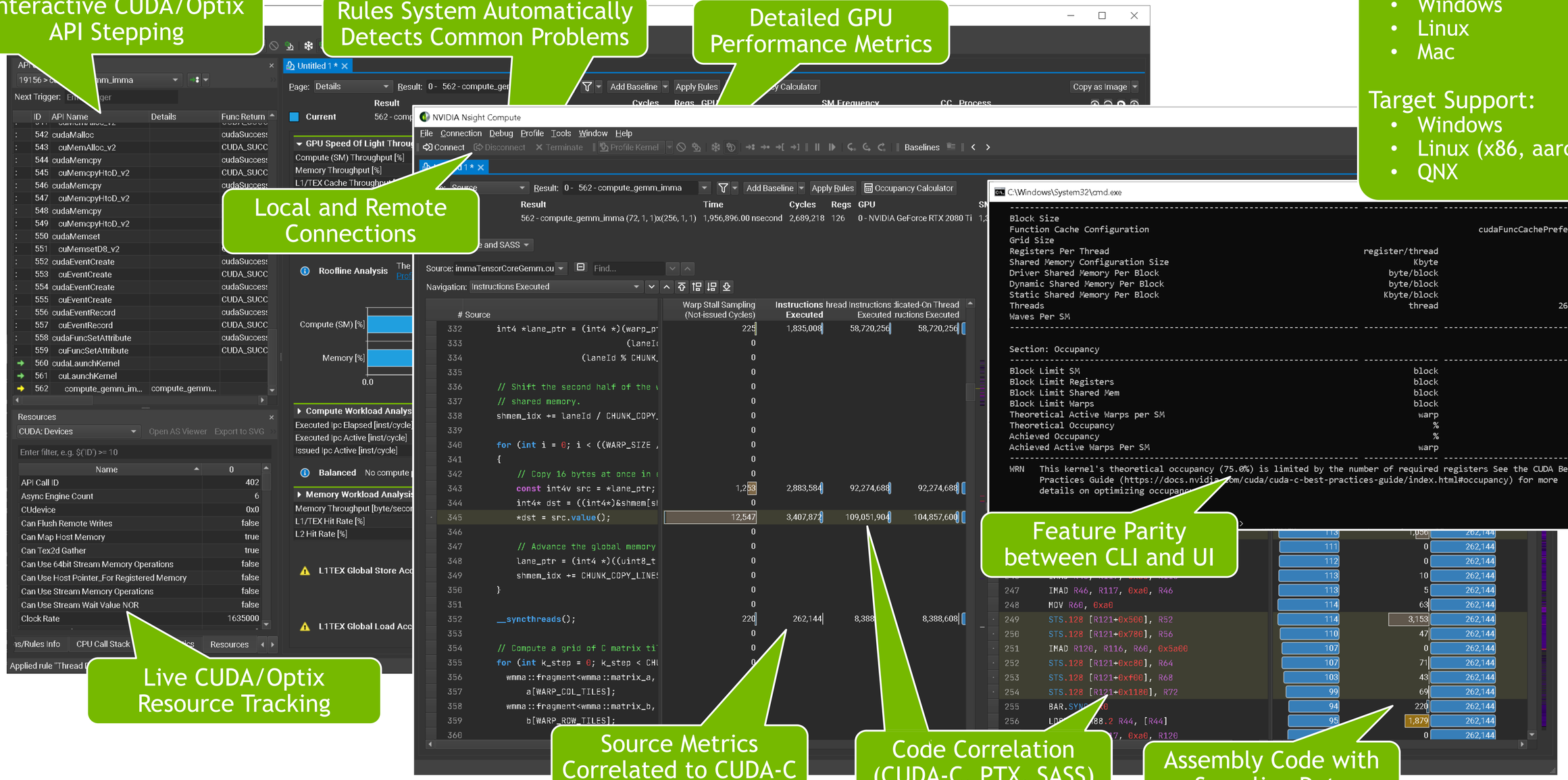
Code Correlation (CUDA-C, PTX, SASS)

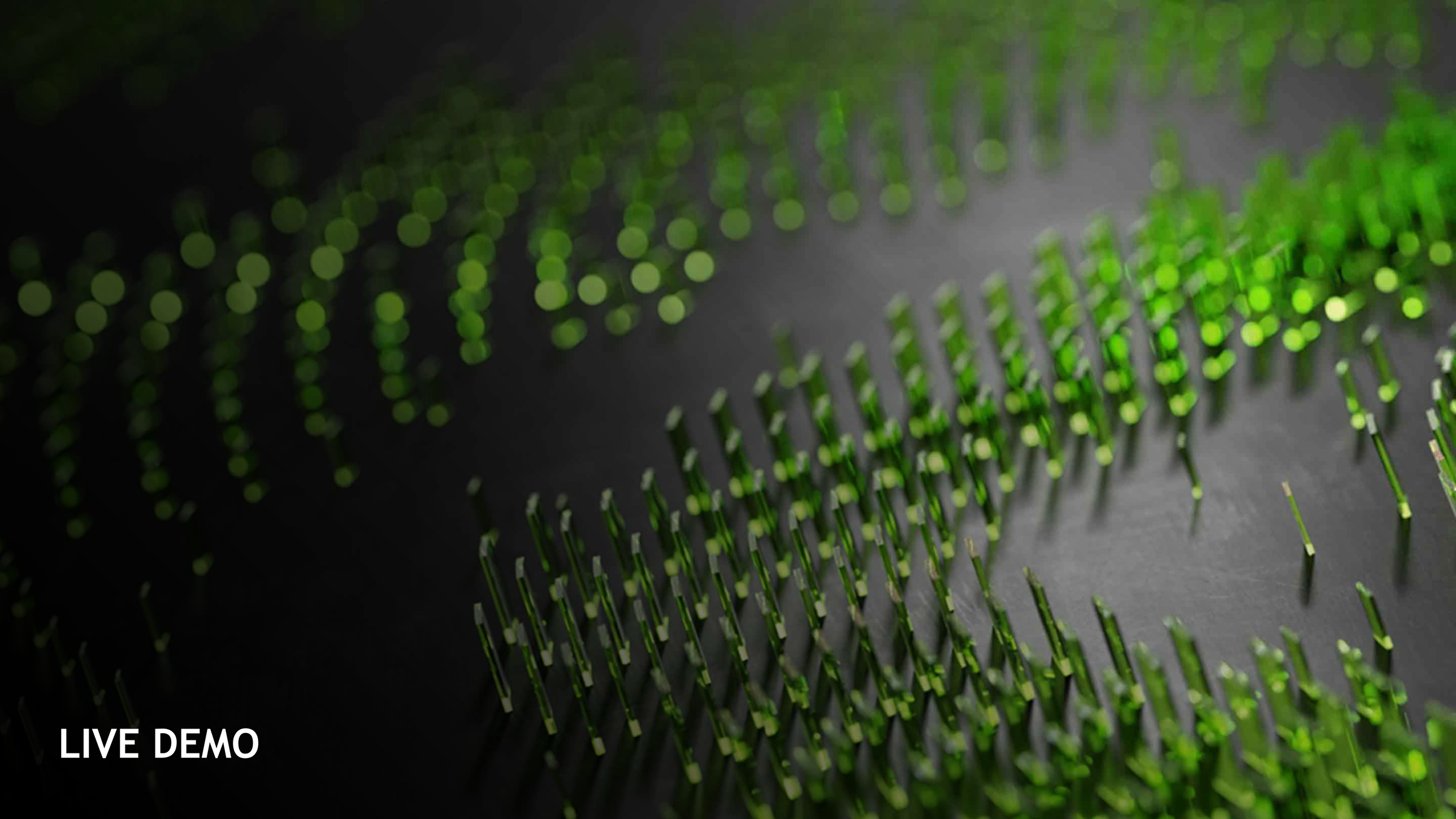
Feature Parity between CLI and UI

Assembly Code with Sampling Data

Host Support:
- Windows
- Linux
- Mac

Target Support:
- Windows
- Linux (x86, aarch64)
- QNX




LIVE DEMO

L1 / SHARED MEMORY

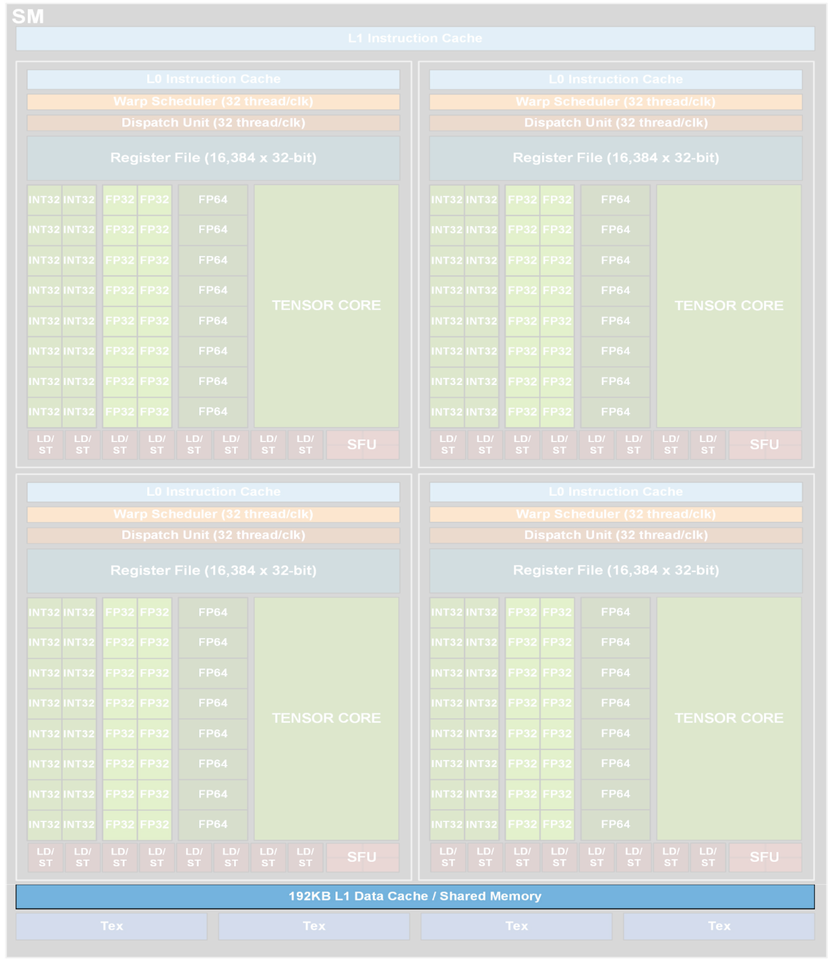
Overview

- Combined L1 Data Cache and Shared Memory
  - 1 L1TEX unit per SM
  - Same physical memory used for L1 cache and shared memory
  - L1 cache uses physical memory as data banks
  - Carveout reserves part of physical memory for shared memory
    - User configurable carveout sizes
  - Similar runtime expectations for shared memory and L1 cache

- Specification of Physical Memory (GA100):
  - Total Size per SM: 192KB
  - Shared Memory Configs: 0, 8, 16, 32, 64, 100, 132, or 164KB
  - Number of Banks: 32
  - Successive 4Byte words map to successive banks
  - 4Byte data access per bank per cycle

L1 / SHARED MEMORY

Overview

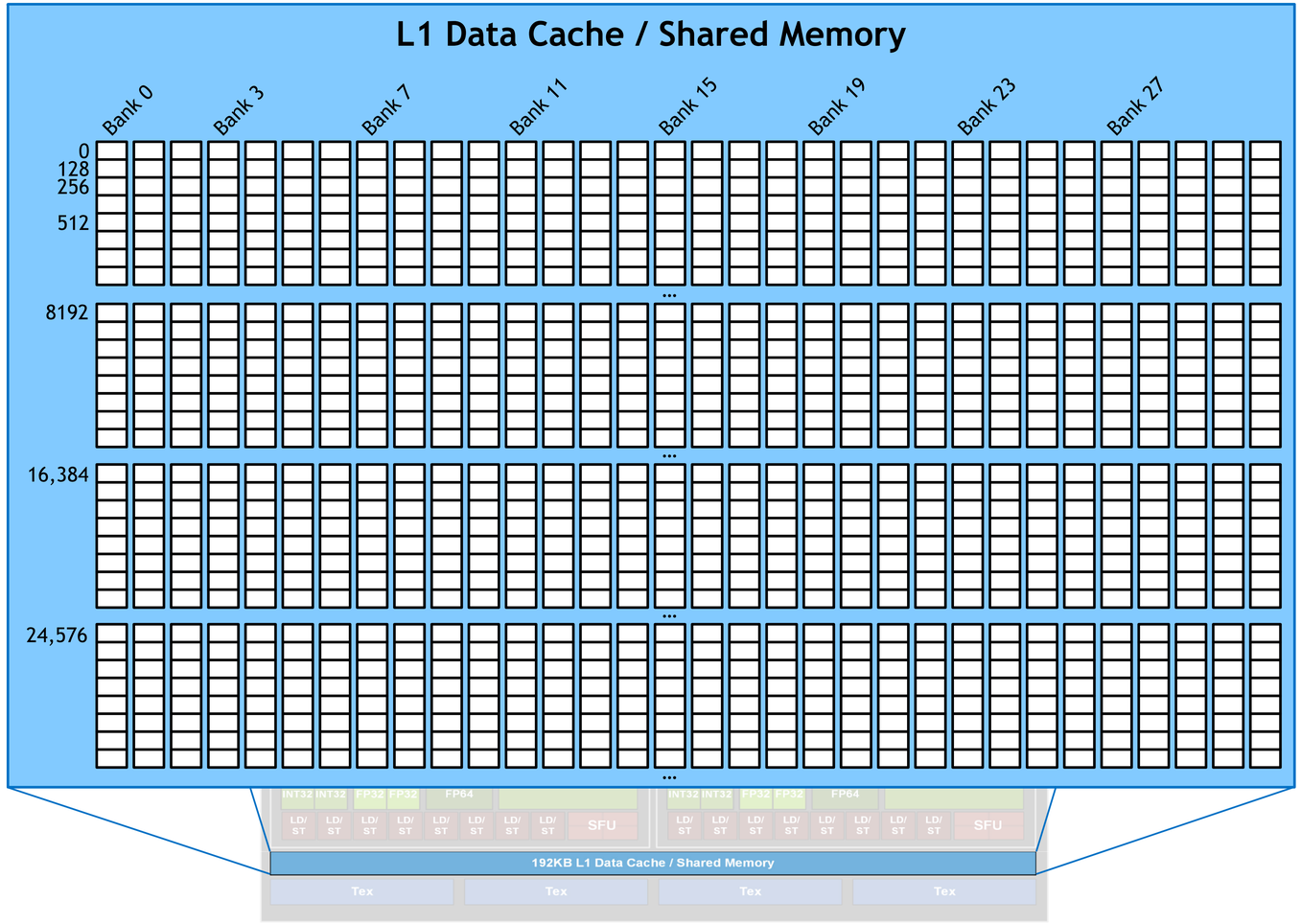


- Combined L1 Data Cache and Shared Memory
  - 1 L1TEX unit per SM
  - Same physical memory used for L1 cache and shared memory
  - L1 cache uses physical memory as data banks
  - Carveout reserves part of physical memory for shared memory
    - User configurable carveout sizes
  - Similar runtime expectations for shared memory and L1 cache

- Specification of Physical Memory (GA100):
  - Total Size per SM: 192KB
  - Shared Memory Configs: 0, 8, 16, 32, 64, 100, 132, or 164KB
  - Number of Banks: 32
  - Successive 4Byte words map to successive banks
  - 4Byte data access per bank per cycle

L1 / SHARED MEMORY

Physical Memory Layout

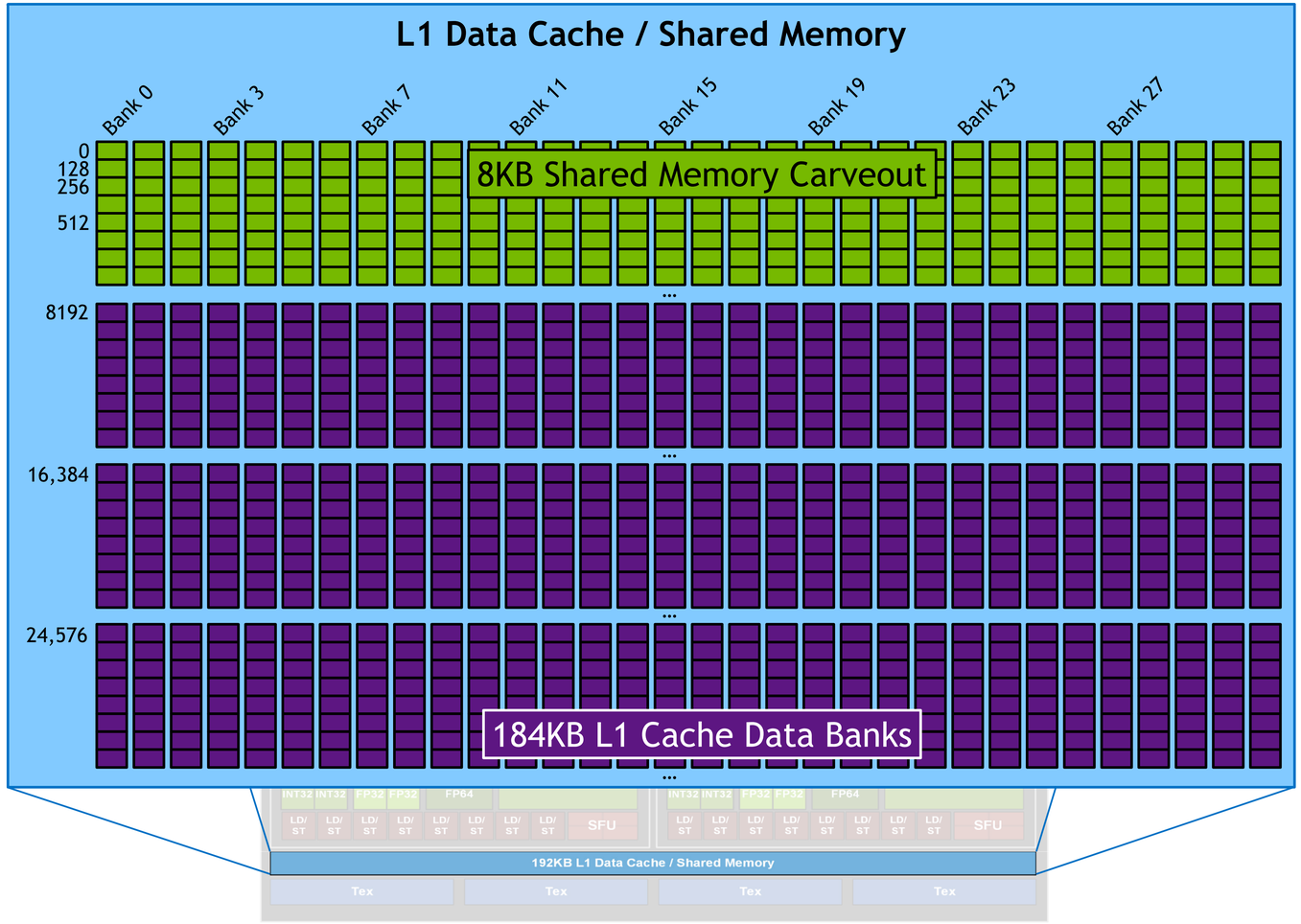


- Combined L1 Data Cache and Shared Memory
  - 1 L1TEX unit per SM
  - Same physical memory used for L1 cache and shared memory
  - L1 cache uses physical memory as data banks
  - Carveout reserves part of physical memory for shared memory
    - User configurable carveout sizes
  - Similar runtime expectations for shared memory and L1 cache

- Specification of Physical Memory (GA100):
  - Total Size per SM: 192KB
  - Shared Memory Configs: 0, 8, 16, 32, 64, 100, 132, or 164KB
  - Number of Banks: 32
  - Successive 4Byte words map to successive banks
  - 4Byte data access per bank per cycle

L1 / SHARED MEMORY

Carveout Example

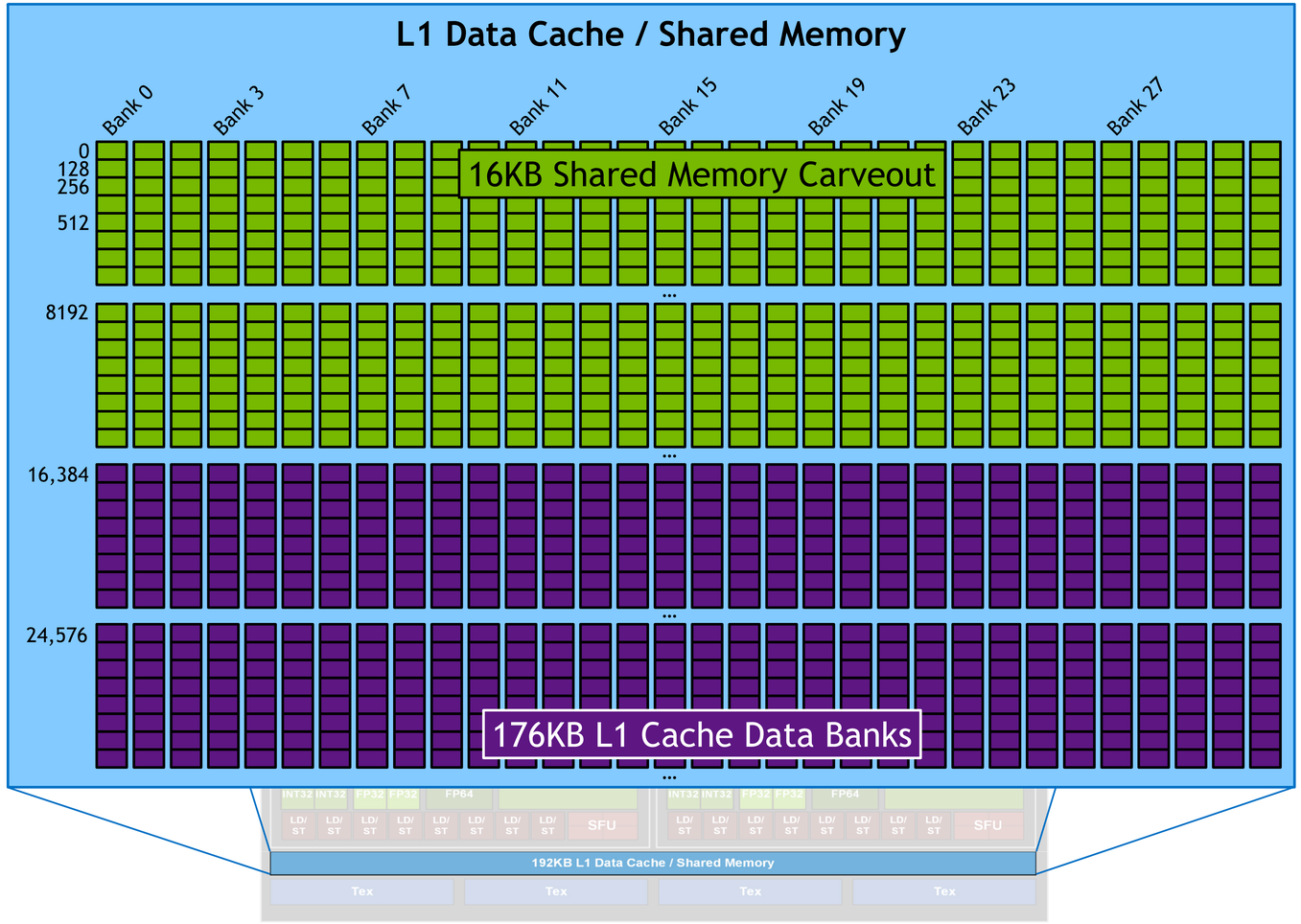


- Combined L1 Data Cache and Shared Memory
  - 1 L1TEX unit per SM
  - Same physical memory used for L1 cache and shared memory
  - L1 cache uses physical memory as data banks
  - Carveout reserves part of physical memory for shared memory
    - User configurable carveout sizes
  - Similar runtime expectations for shared memory and L1 cache

- Specification of Physical Memory (GA100):
  - Total Size per SM: 192KB
  - Shared Memory Configs: 0, 8, 16, 32, 64, 100, 132, or 164KB
  - Number of Banks: 32
  - Successive 4Byte words map to successive banks
  - 4Byte data access per bank per cycle

L1 / SHARED MEMORY

Carveout Example

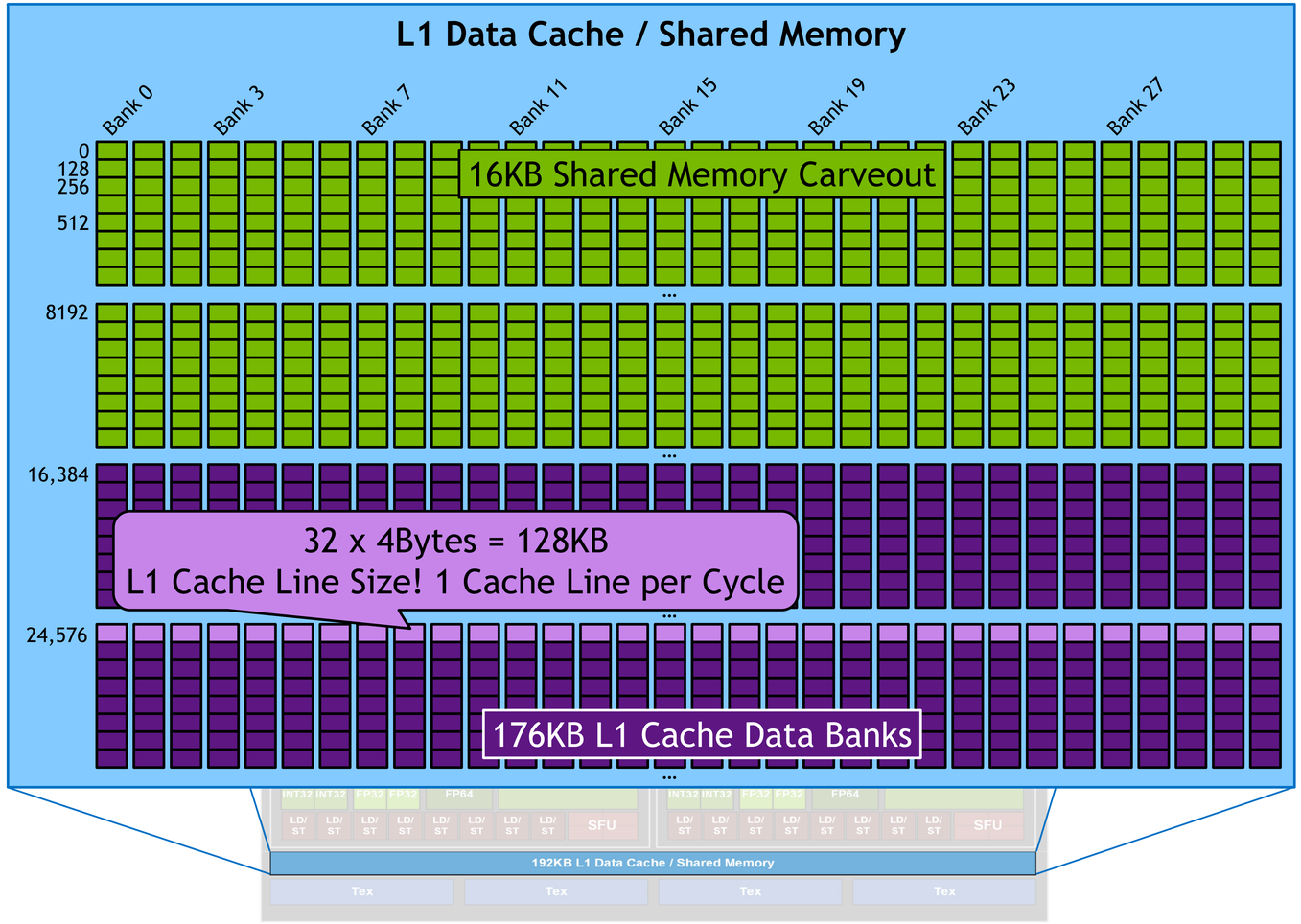


- Combined L1 Data Cache and Shared Memory
  - 1 L1TEX unit per SM
  - Same physical memory used for L1 cache and shared memory
  - L1 cache uses physical memory as data banks
  - Carveout reserves part of physical memory for shared memory
    - User configurable carveout sizes
  - Similar runtime expectations for shared memory and L1 cache

- Specification of Physical Memory (GA100):
  - Total Size per SM: 192KB
  - Shared Memory Configs: 0, 8, 16, 32, 64, 100, 132, or 164KB
  - Number of Banks: 32
  - Successive 4Byte words map to successive banks
  - 4Byte data access per bank per cycle

L1 / SHARED MEMORY

L1 Cache Perspective

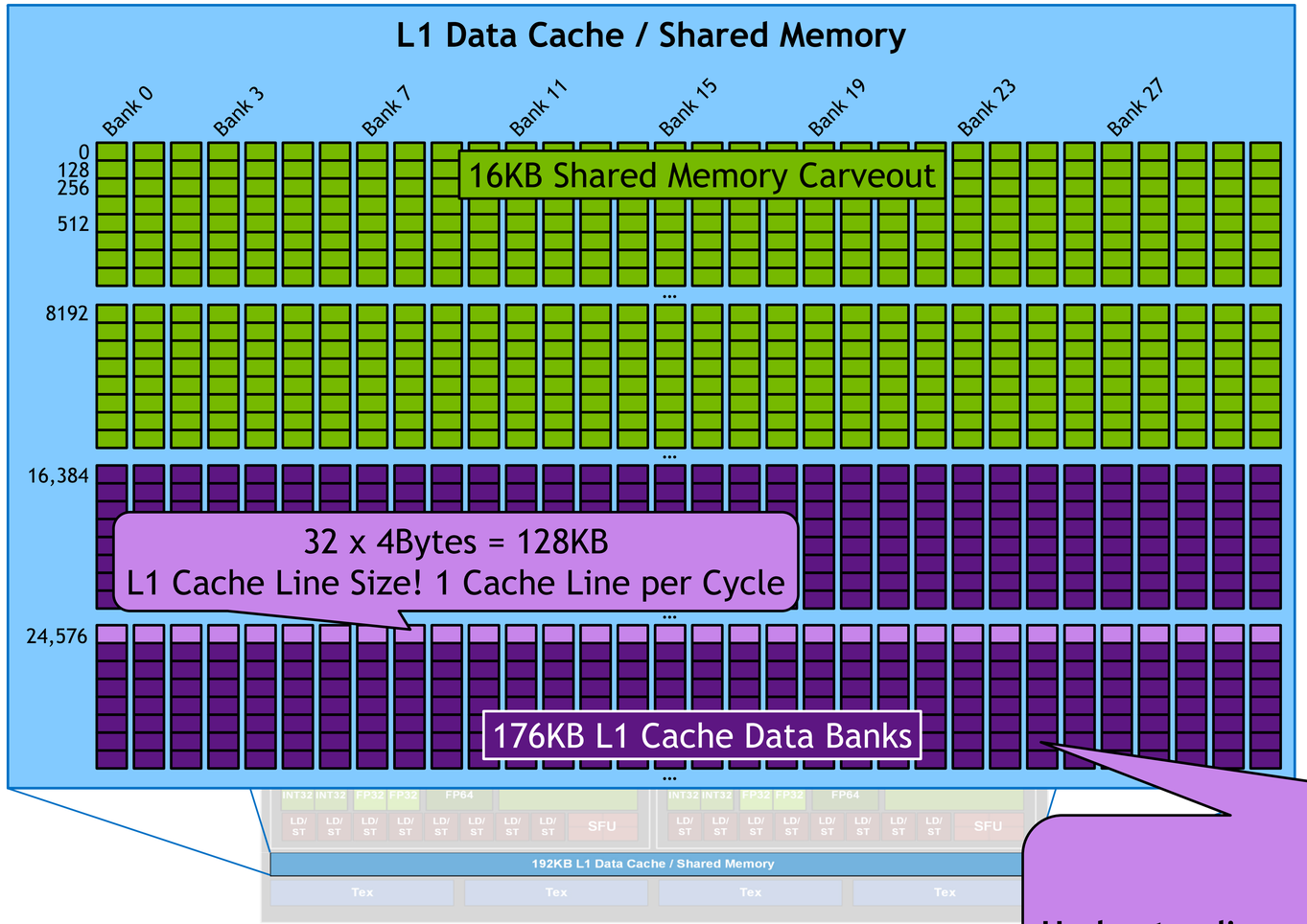


- Combined L1 Data Cache and Shared Memory
  - 1 L1TEX unit per SM
  - Same physical memory used for L1 cache and shared memory
  - L1 cache uses physical memory as data banks
  - Carveout reserves part of physical memory for shared memory
    - User configurable carveout sizes
  - Similar runtime expectations for shared memory and L1 cache

- Specification of Physical Memory (GA100):
  - Total Size per SM: 192KB
  - Shared Memory Configs: 0, 8, 16, 32, 64, 100, 132, or 164KB
  - Number of Banks: 32
  - Successive 4Byte words map to successive banks
  - 4Byte data access per bank per cycle

L1 / SHARED MEMORY

L1 Cache Perspective



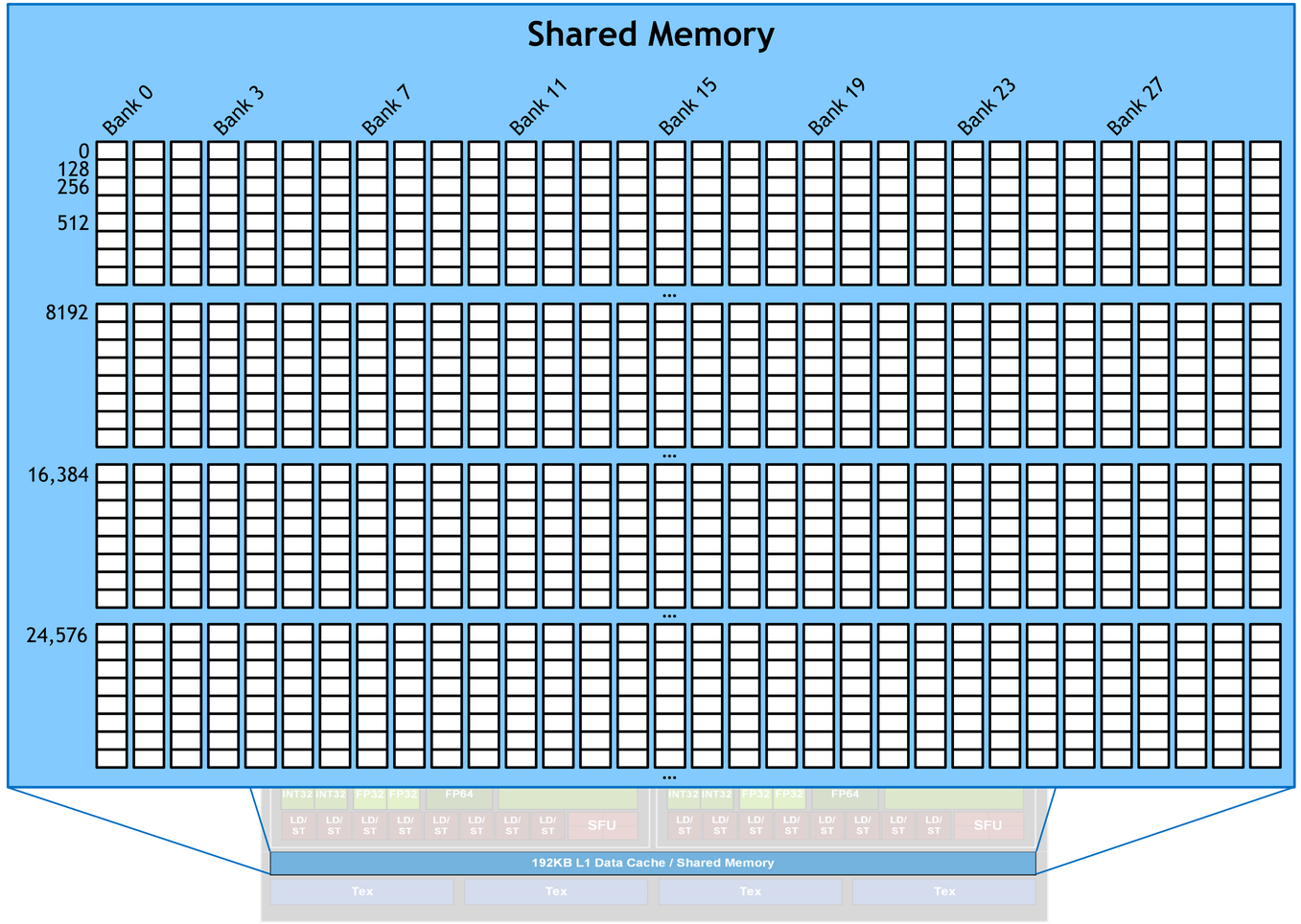
- Combined L1 Data Cache and Shared Memory
  - 1 L1TEX unit per SM
  - Same physical memory used for L1 cache and shared memory
  - L1 cache uses physical memory as data banks
  - Carveout reserves part of physical memory for shared memory
    - User configurable carveout sizes
  - Similar runtime expectations for shared memory and L1 cache

- Specification of Physical Memory (GA100):
  - Total Size per SM: 192KB
  - Shared Memory Configs: 0, 8, 16, 32, 64, 100, 132, or 164KB
  - Number of Banks: 32
  - Successive 4Byte words map to successive banks
  - 4Byte data access per bank per cycle

For More Details:
Requests, Wavefronts, Sectors Metrics:
Understanding and Optimizing Memory-Bound Kernels with Nsight Compute
GTC2021
https://www.nvidia.com/en-us/on-demand/session/gtcspring21-s32089/

L1 / SHARED MEMORY

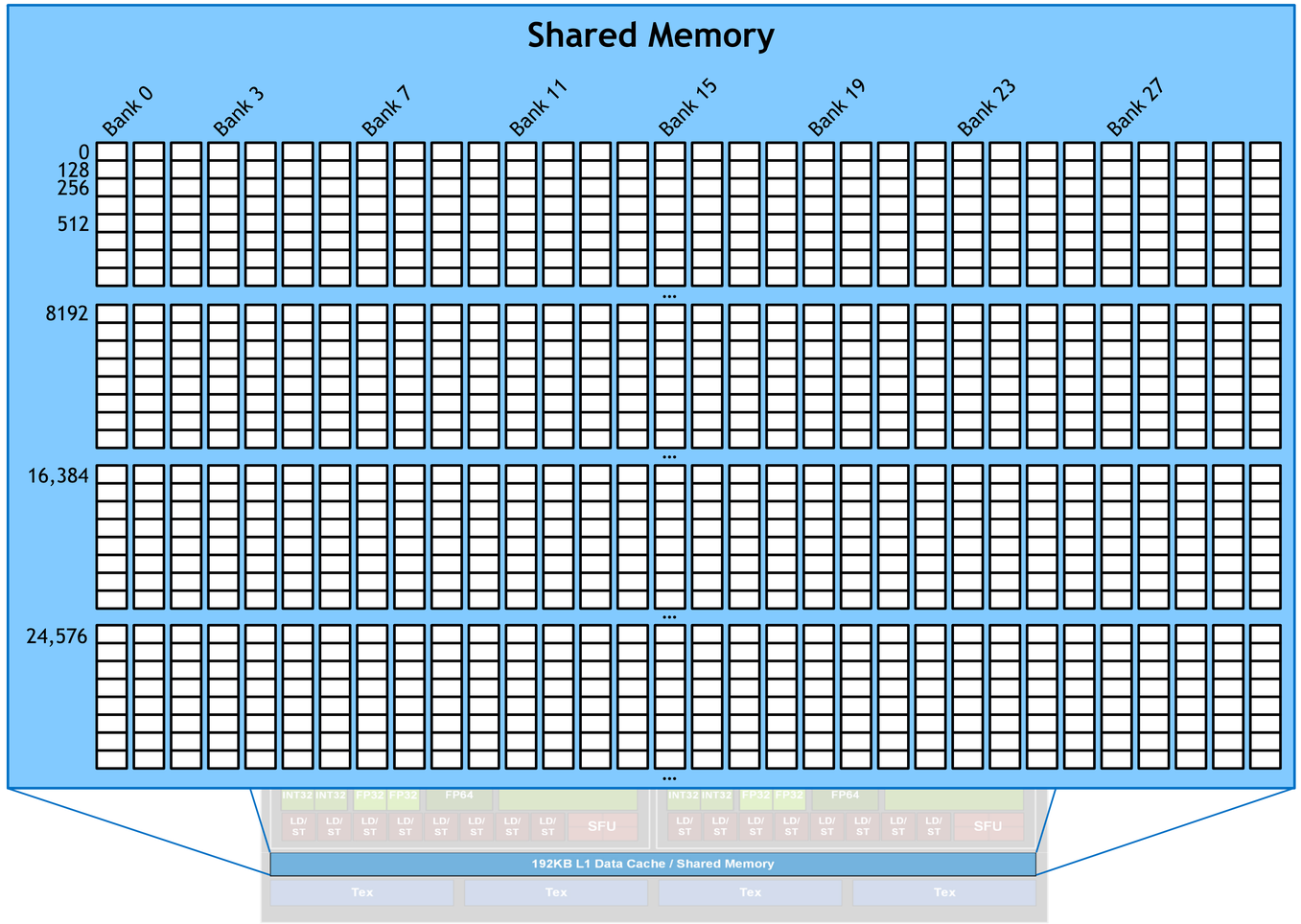
Shared Memory Perspective



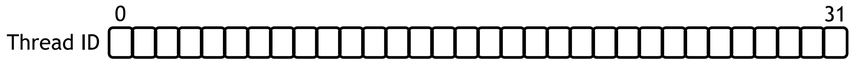
- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

L1 / SHARED MEMORY

Shared Memory Perspective



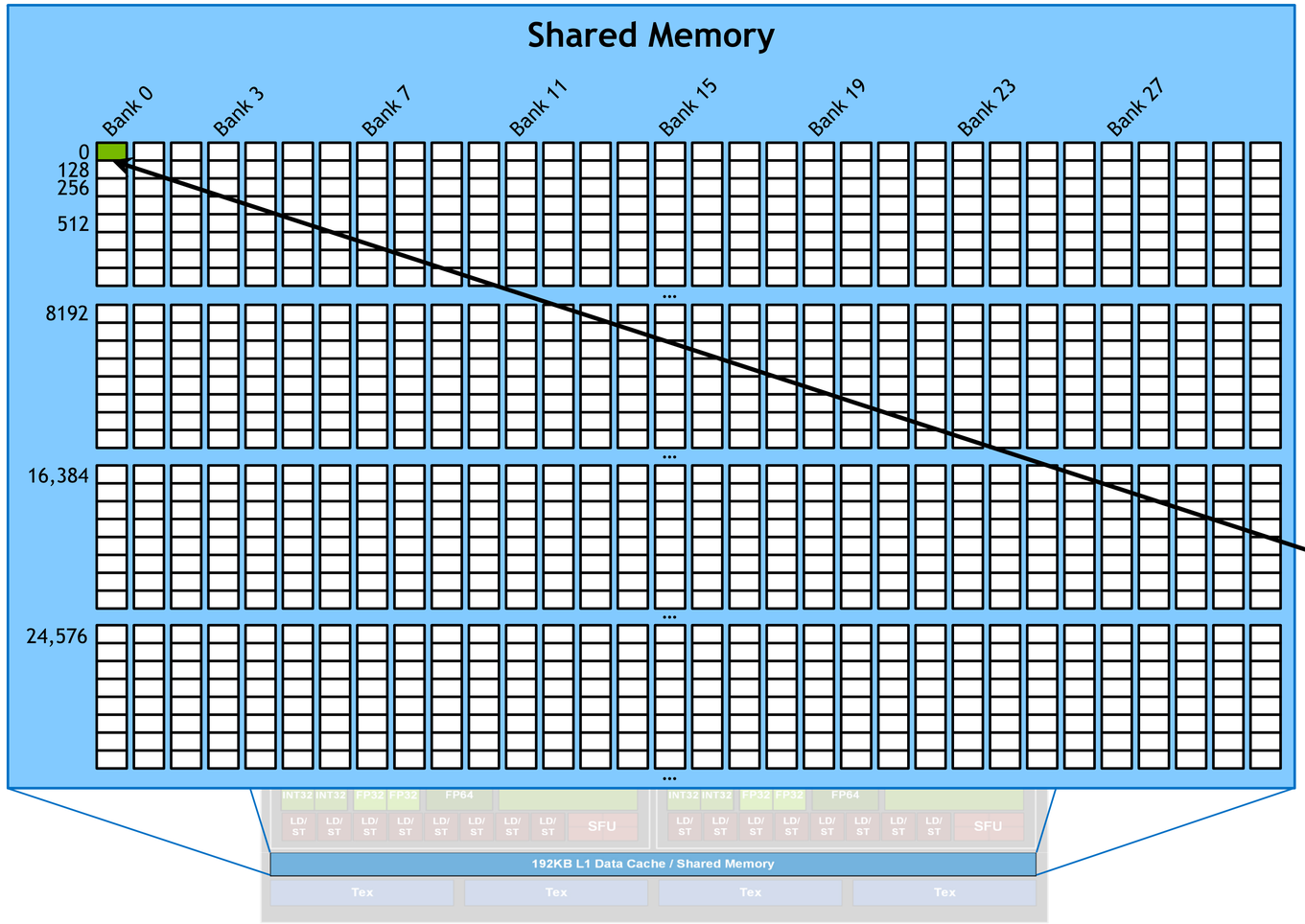
- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts



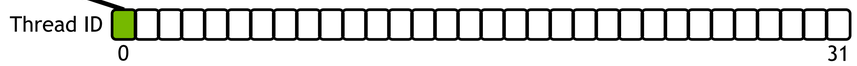
- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive coalesced access (4B stride)

L1 / SHARED MEMORY

Shared Memory Perspective



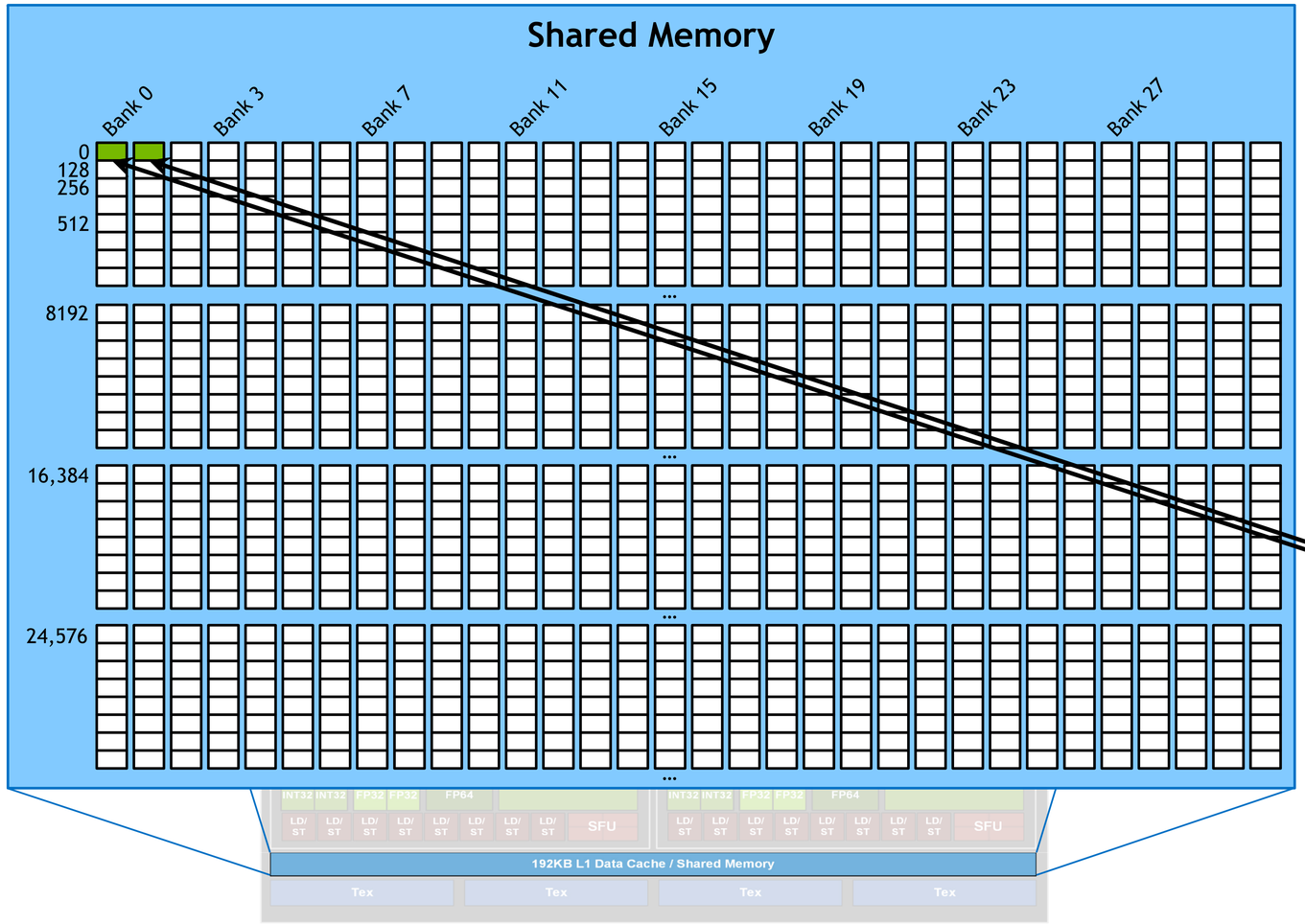
- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts



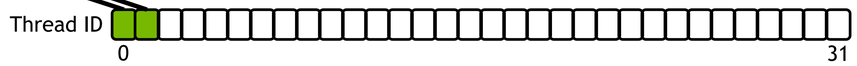
- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive coalesced access (4B stride)

L1 / SHARED MEMORY

Shared Memory Perspective



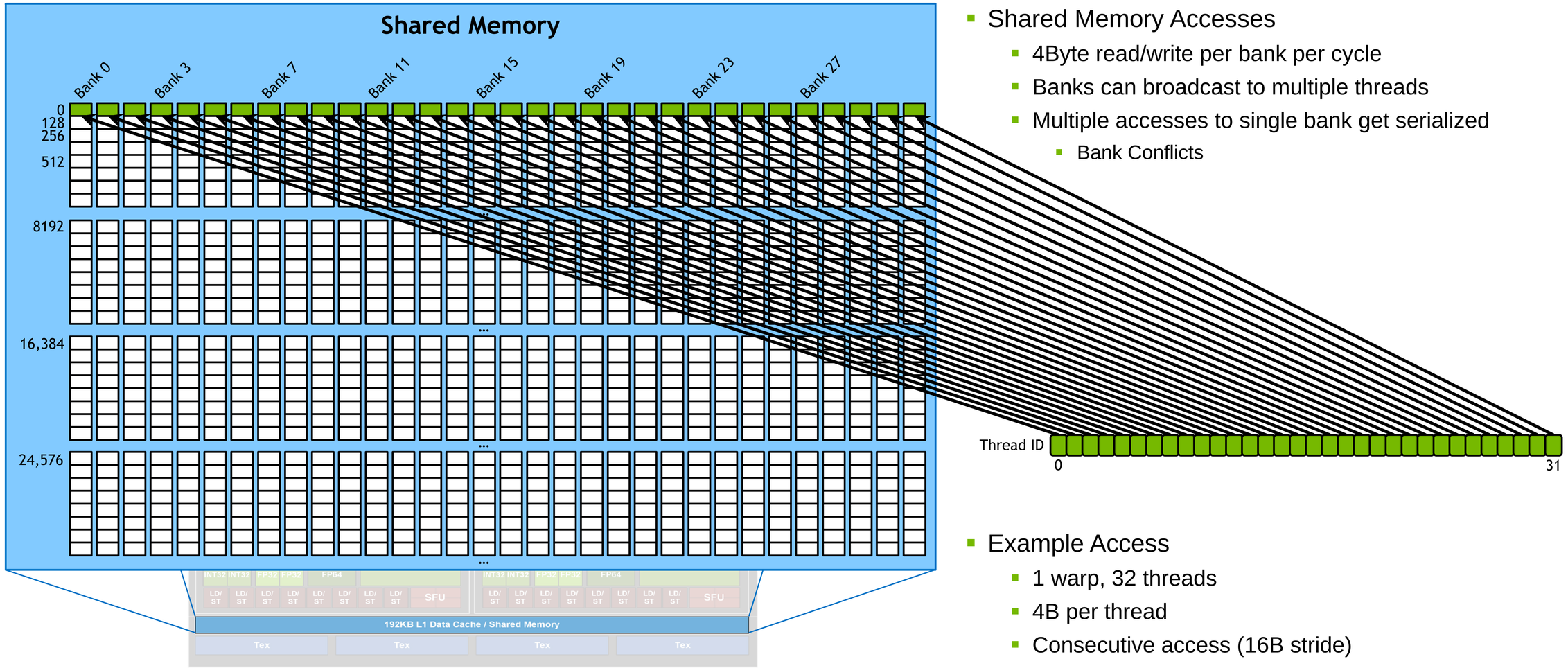
- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts



- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive coalesced access (4B stride)

L1 / SHARED MEMORY

Shared Memory Perspective

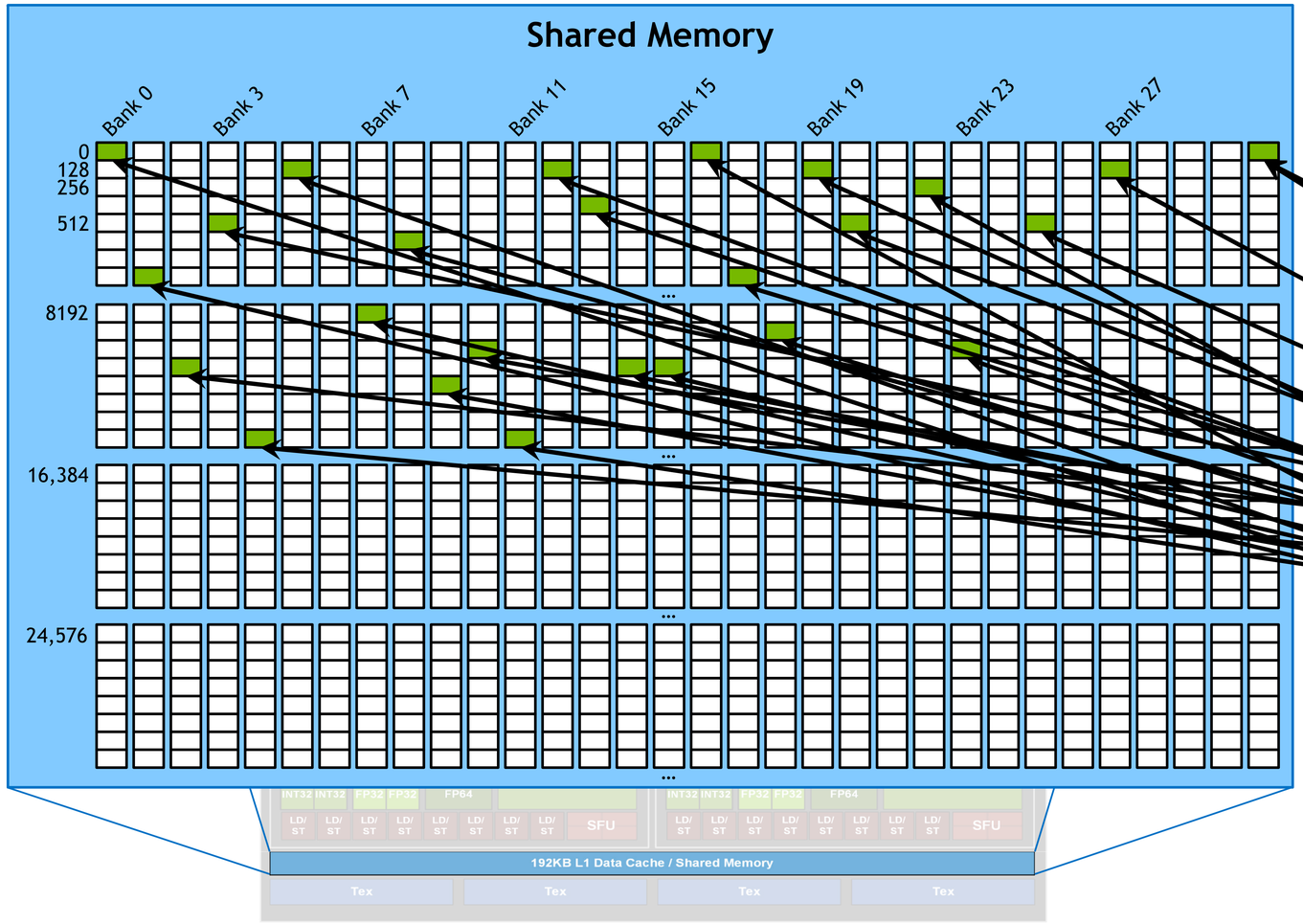


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive access (16B stride)
  - No Conflicts
  - 1 Cycle to access

L1 / SHARED MEMORY

Shared Memory Perspective

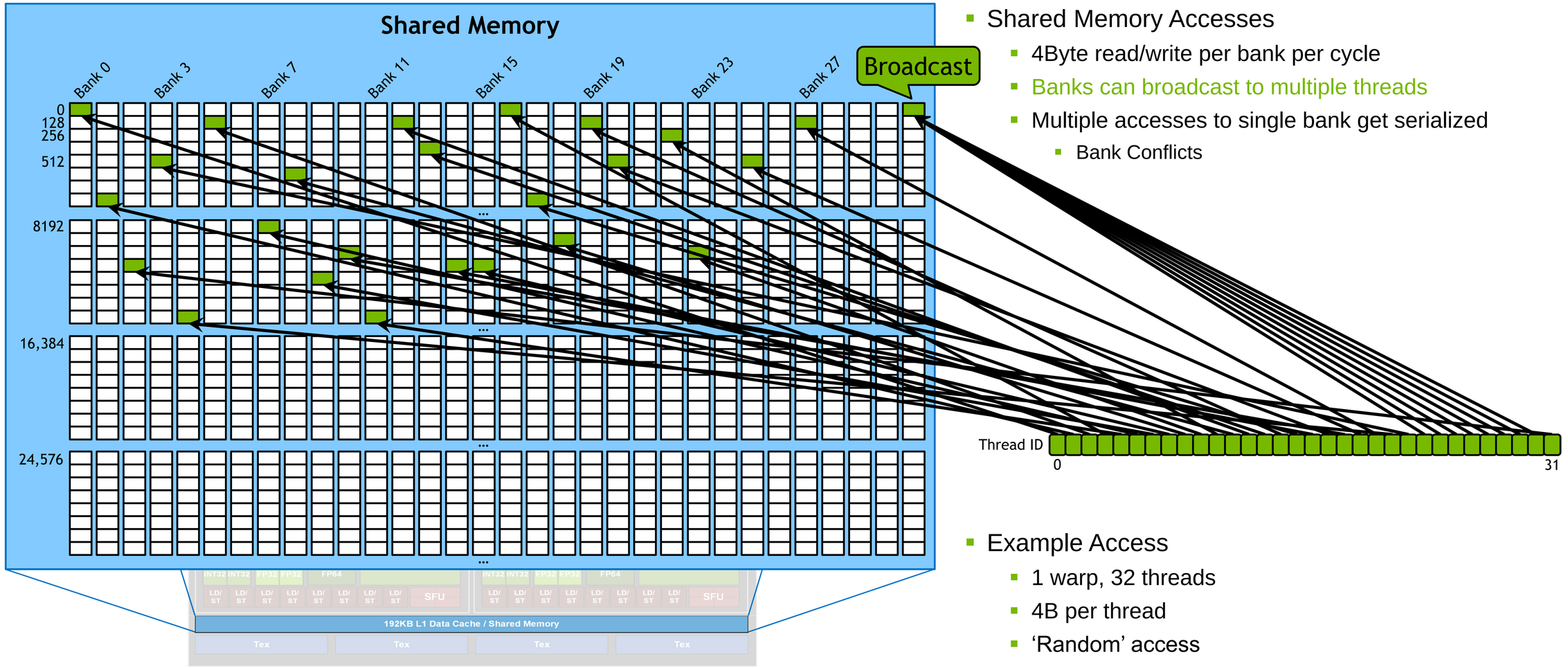


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - ‘Random’ access
  - No Conflicts
  - 1 Cycle to access

L1 / SHARED MEMORY

Shared Memory Perspective

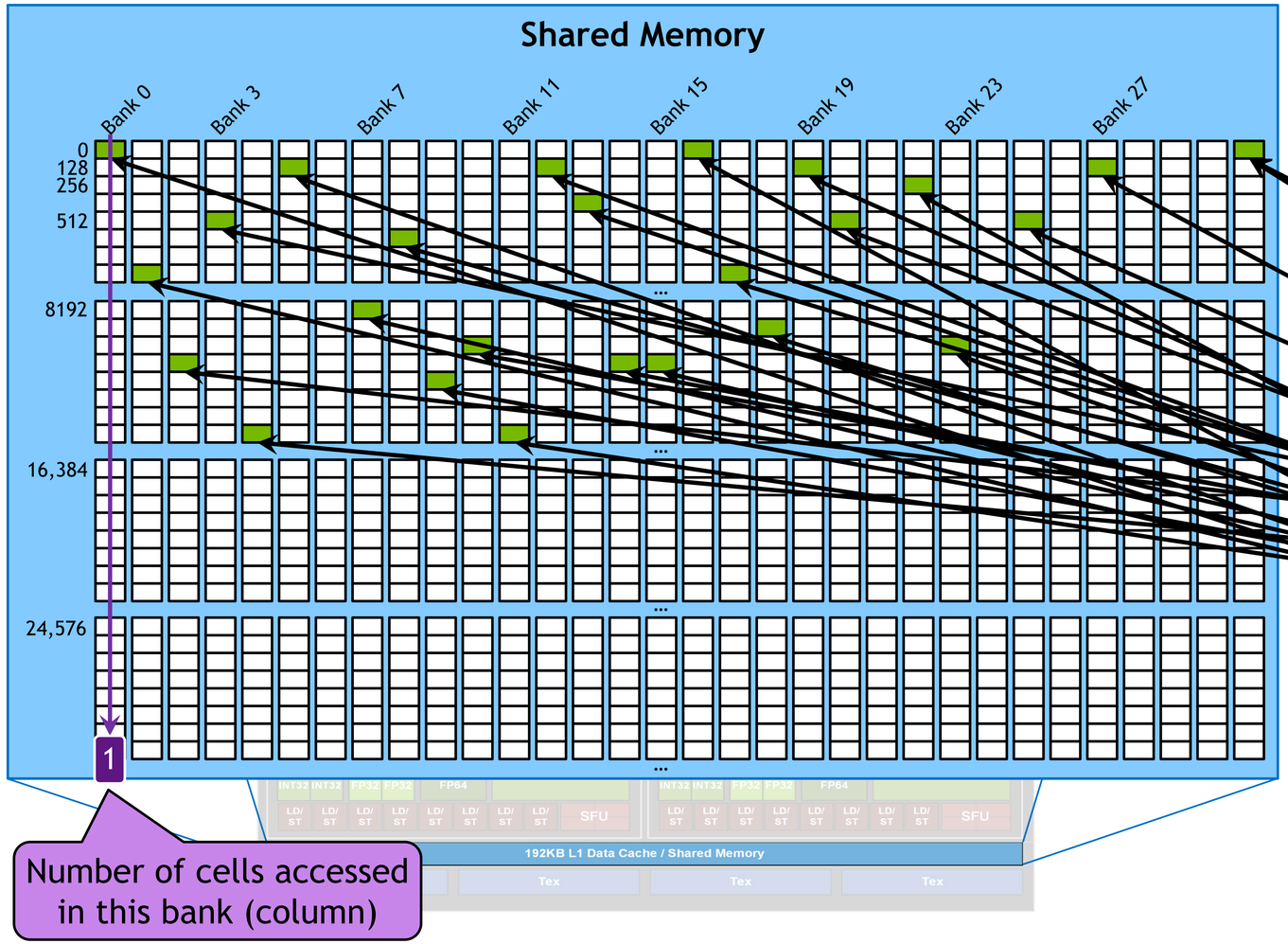


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - ‘Random’ access
  - No Conflicts
  - 1 Cycle to access

L1 / SHARED MEMORY

Shared Memory Perspective

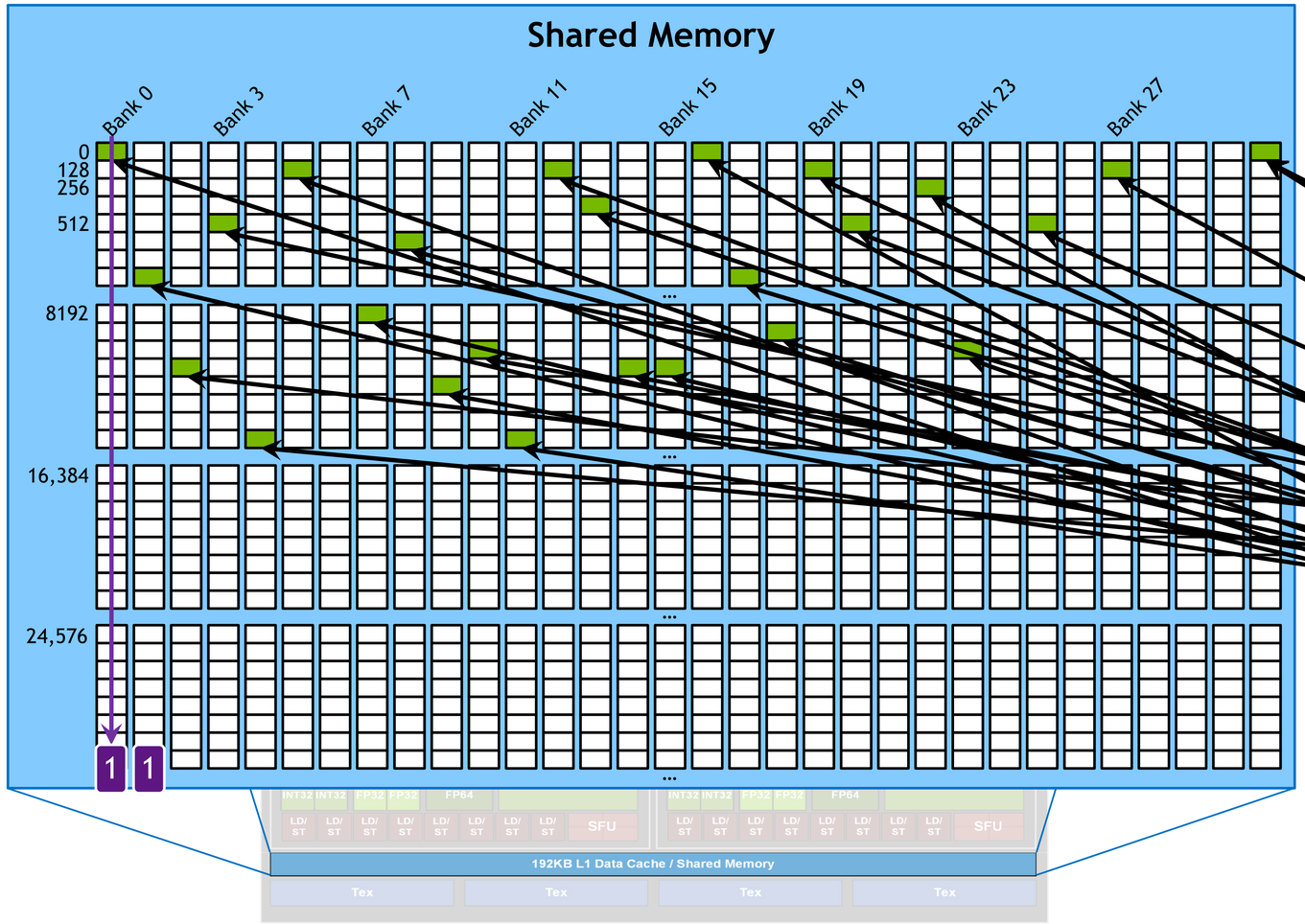


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - ‘Random’ access
  - No Conflicts
  - 1 Cycle to access

L1 / SHARED MEMORY

Shared Memory Perspective

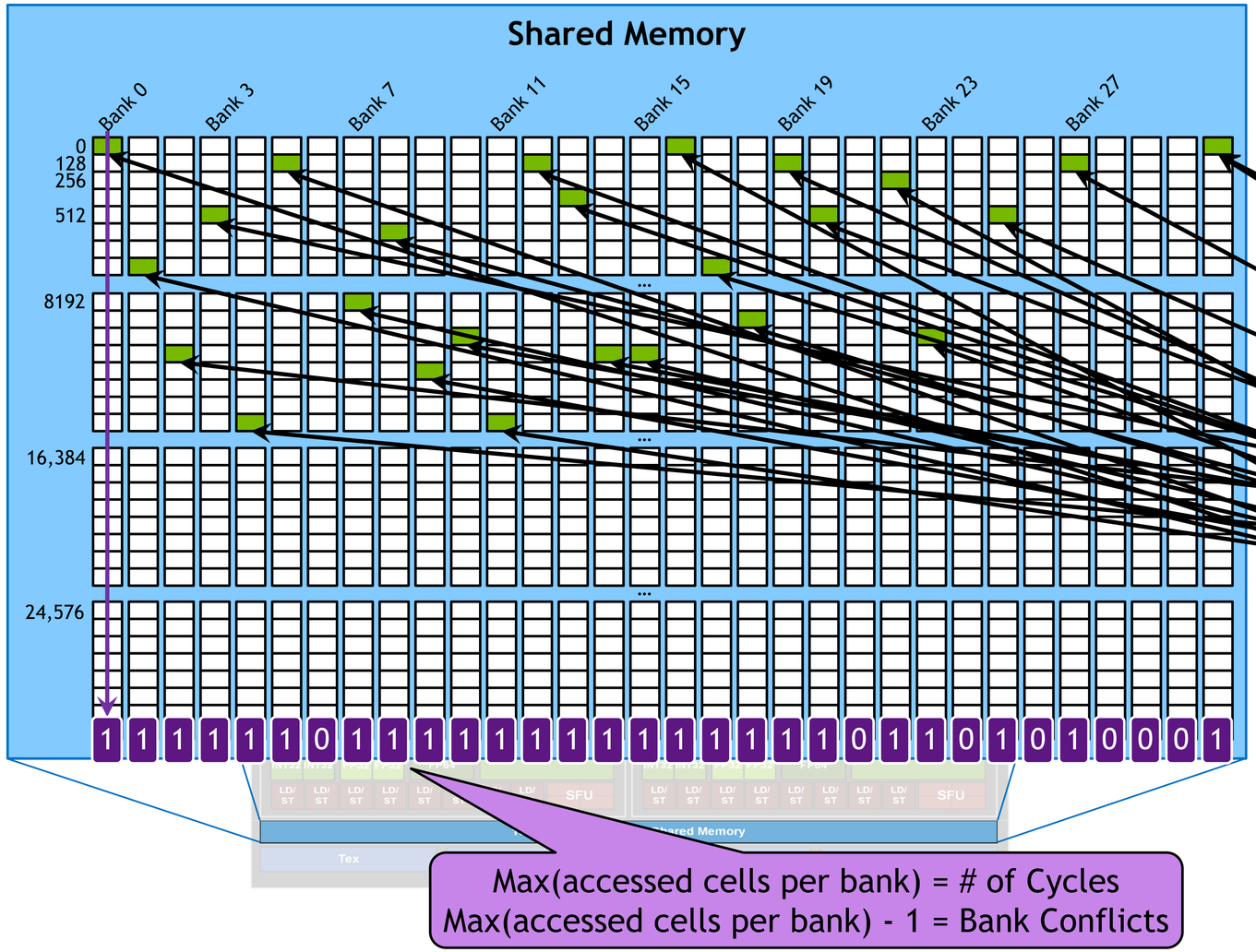


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - ‘Random’ access
  - No Conflicts
  - 1 Cycle to access

L1 / SHARED MEMORY

Shared Memory Perspective

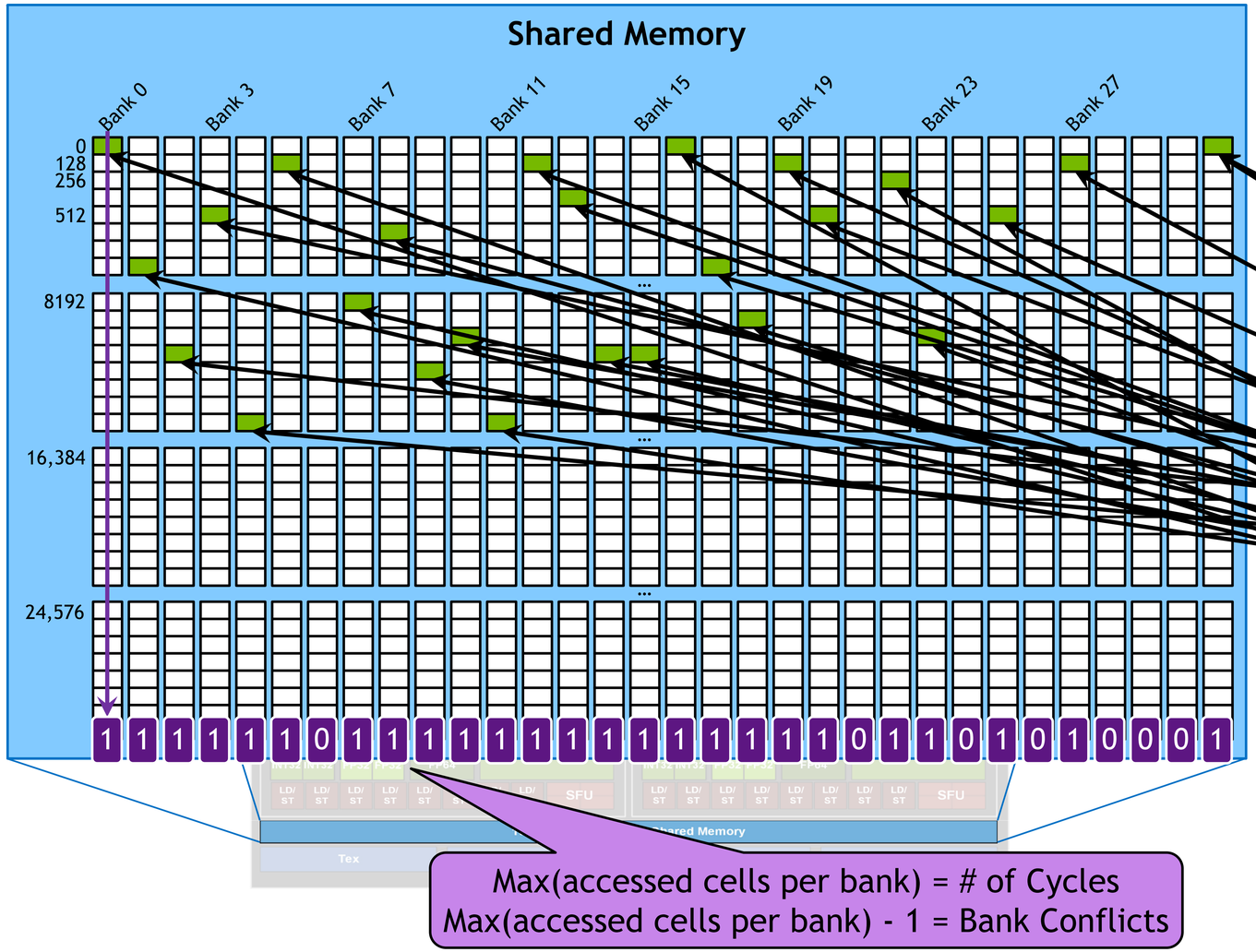


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - ‘Random’ access
  - No Conflicts
  - 1 Cycle to access

L1 / SHARED MEMORY

Shared Memory Perspective



- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

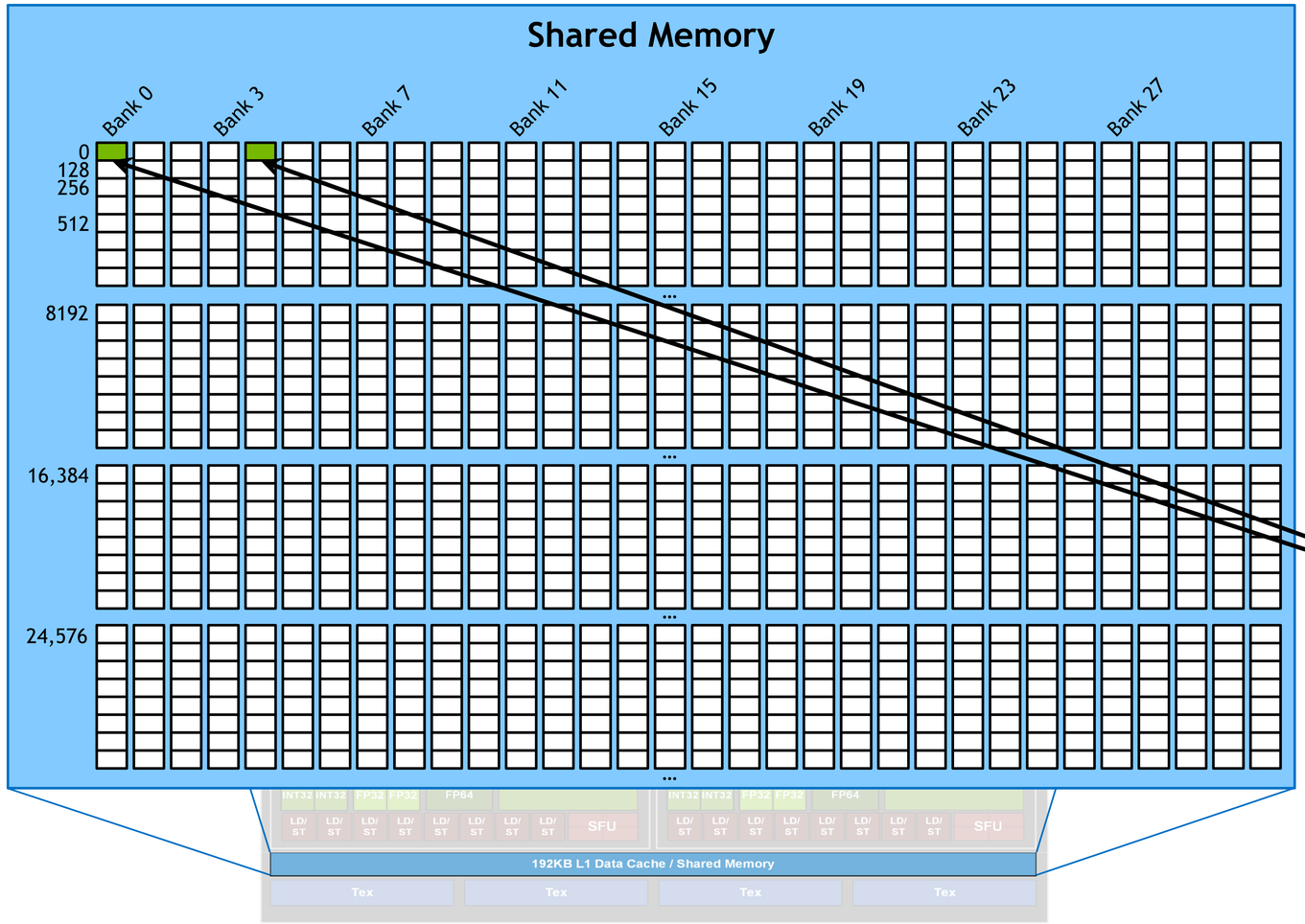
- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - ‘Random’ access
  - No Conflicts
  - 1 Cycle to access

Same Access Pattern to L1 Cache:

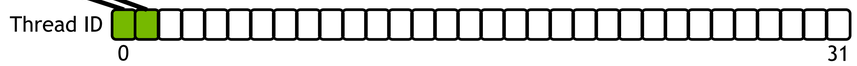
- 13 L1 Cache Lines
- 24 L1 Sector Accesses

L1 / SHARED MEMORY

Shared Memory Perspective



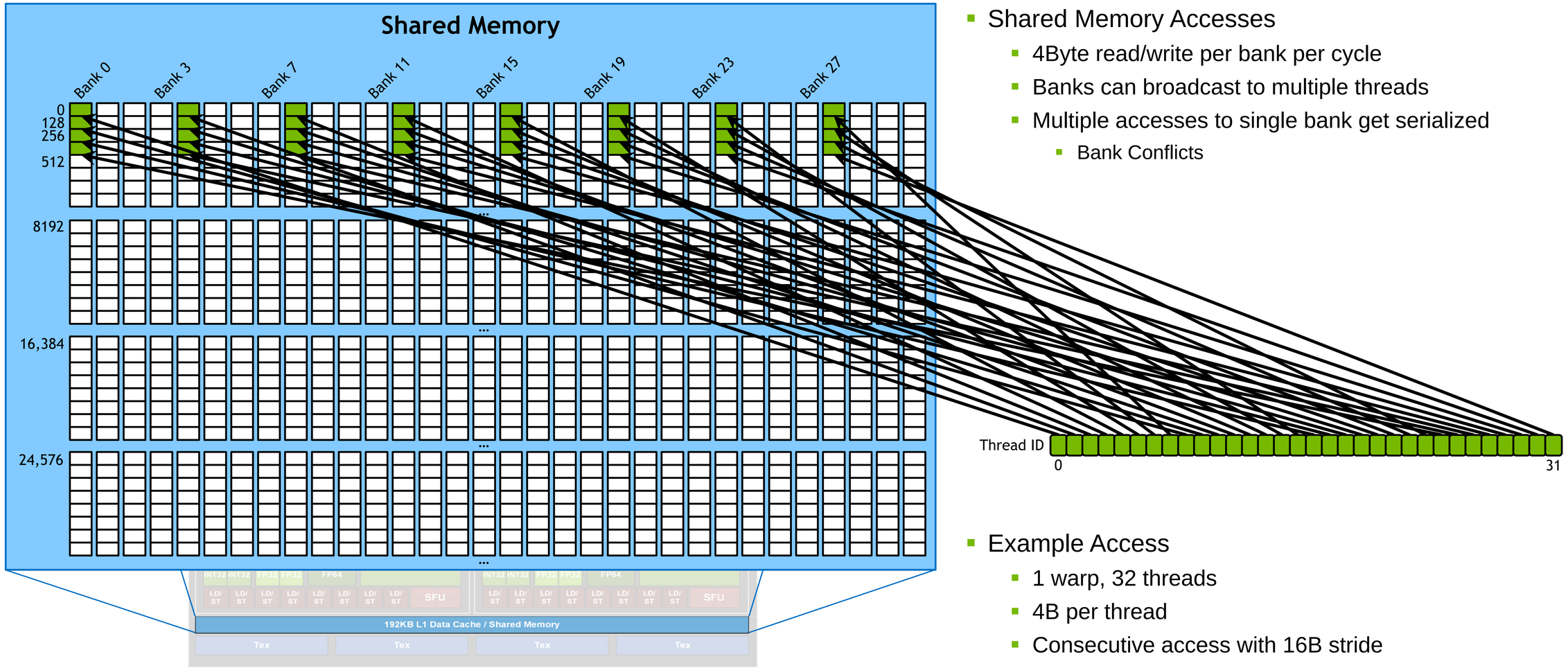
- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts



- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive access with 16B stride

L1 / SHARED MEMORY

Shared Memory Perspective

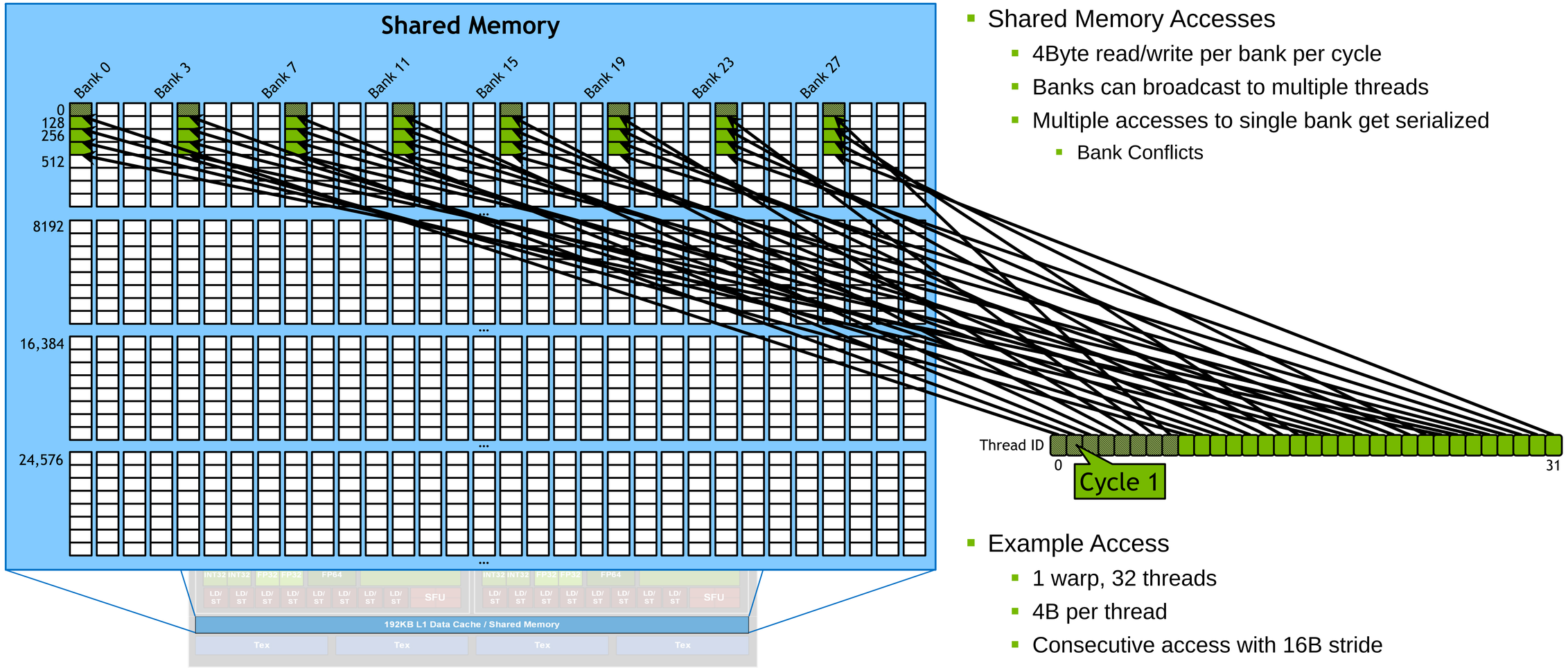


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive access with 16B stride

L1 / SHARED MEMORY

Shared Memory Perspective

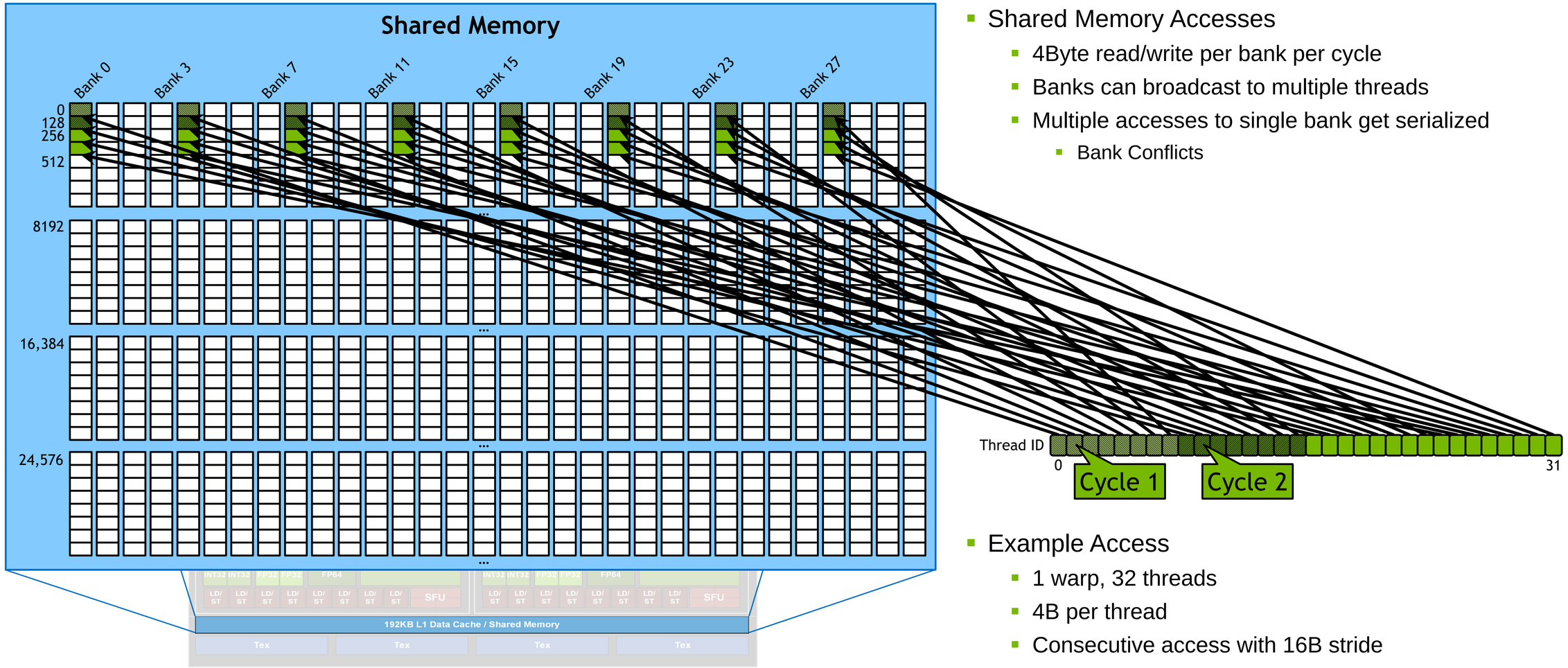


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive access with 16B stride

L1 / SHARED MEMORY

Shared Memory Perspective

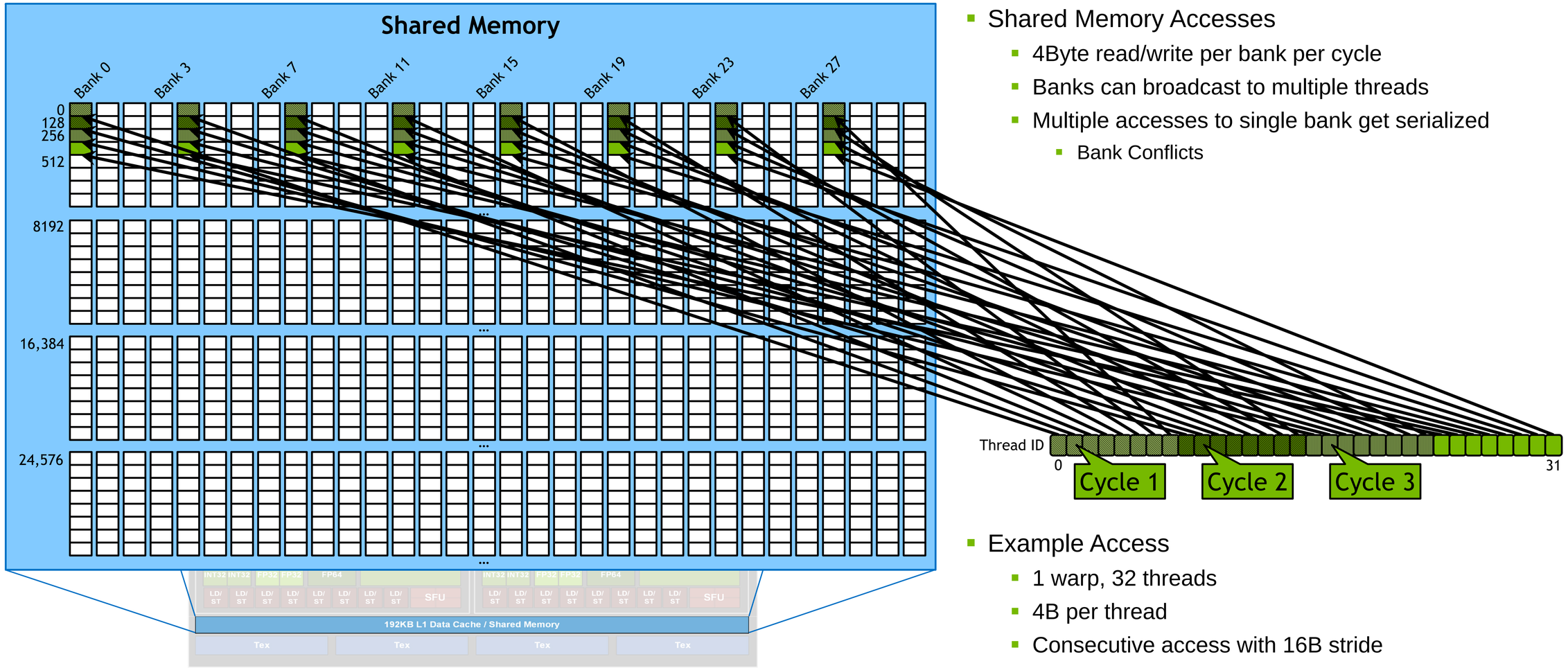


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive access with 16B stride

L1 / SHARED MEMORY

Shared Memory Perspective

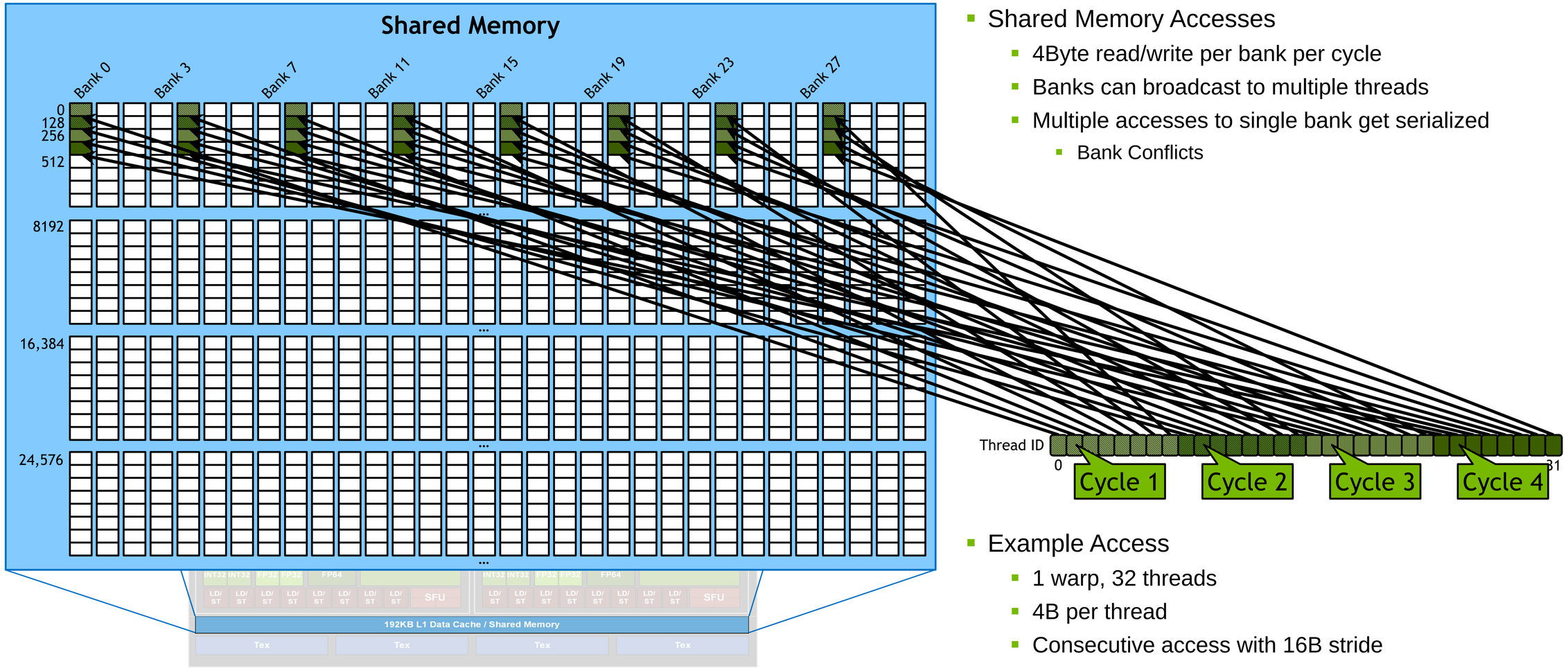


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive access with 16B stride

L1 / SHARED MEMORY

Shared Memory Perspective

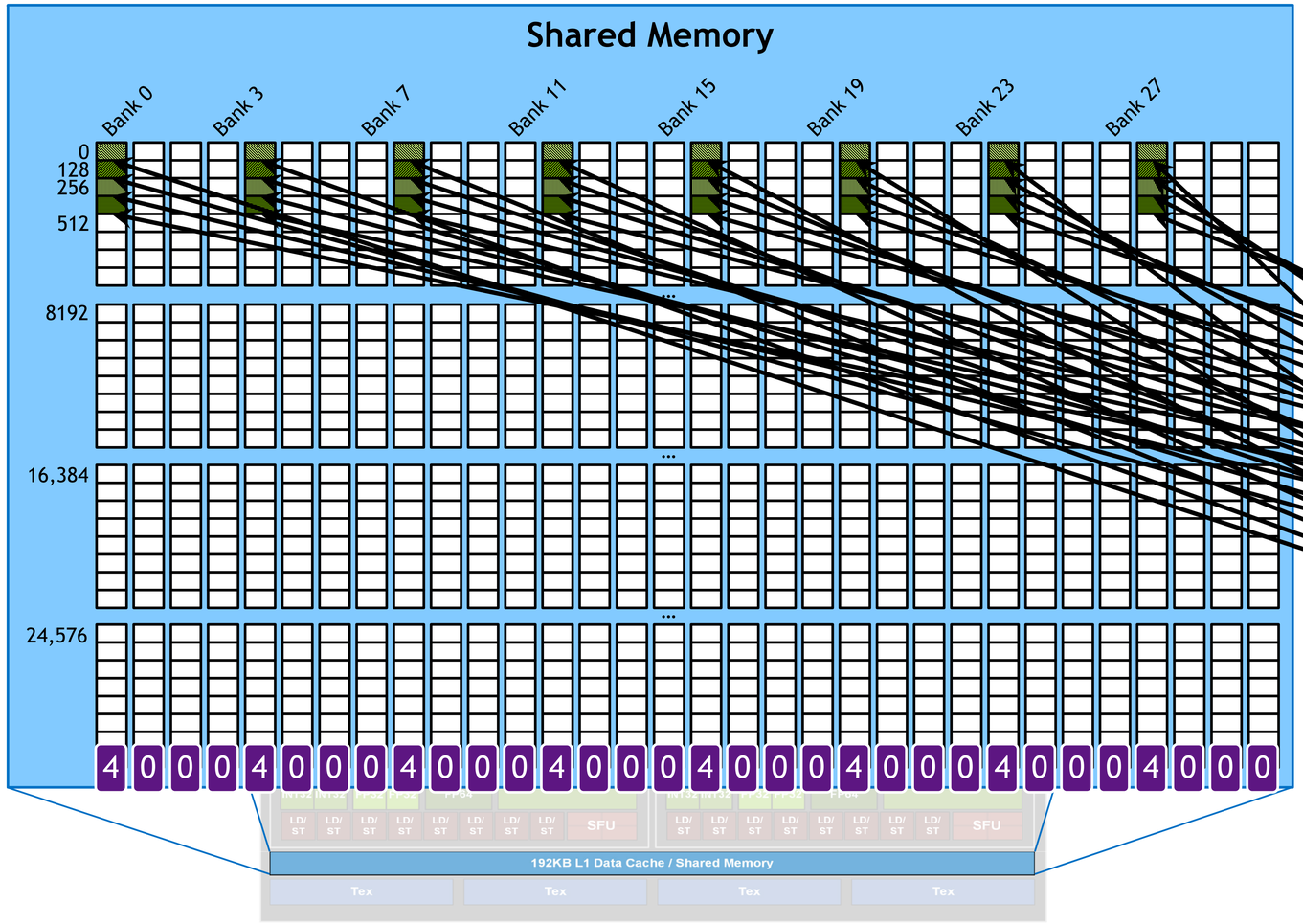


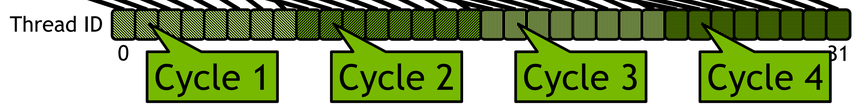
- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive access with 16B stride

L1 / SHARED MEMORY

Shared Memory Perspective



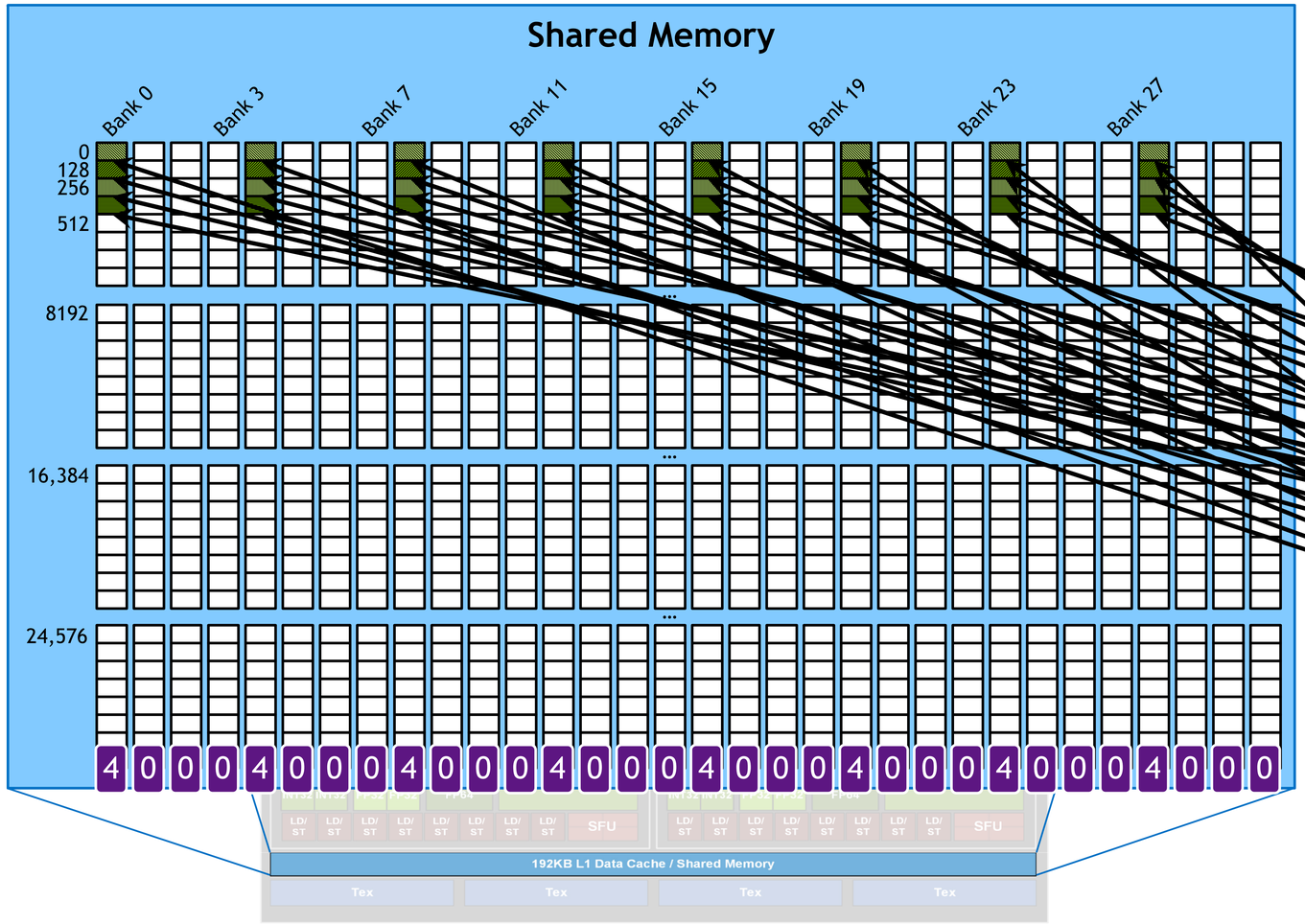


- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

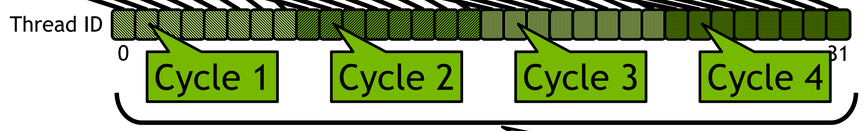
- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive access with 16B stride
  - 3 Bank Conflicts
  - 4 Cycle to access

L1 / SHARED MEMORY

Shared Memory Perspective



- Shared Memory Accesses
  - 4Byte read/write per bank per cycle
  - Banks can broadcast to multiple threads
  - Multiple accesses to single bank get serialized
    - Bank Conflicts

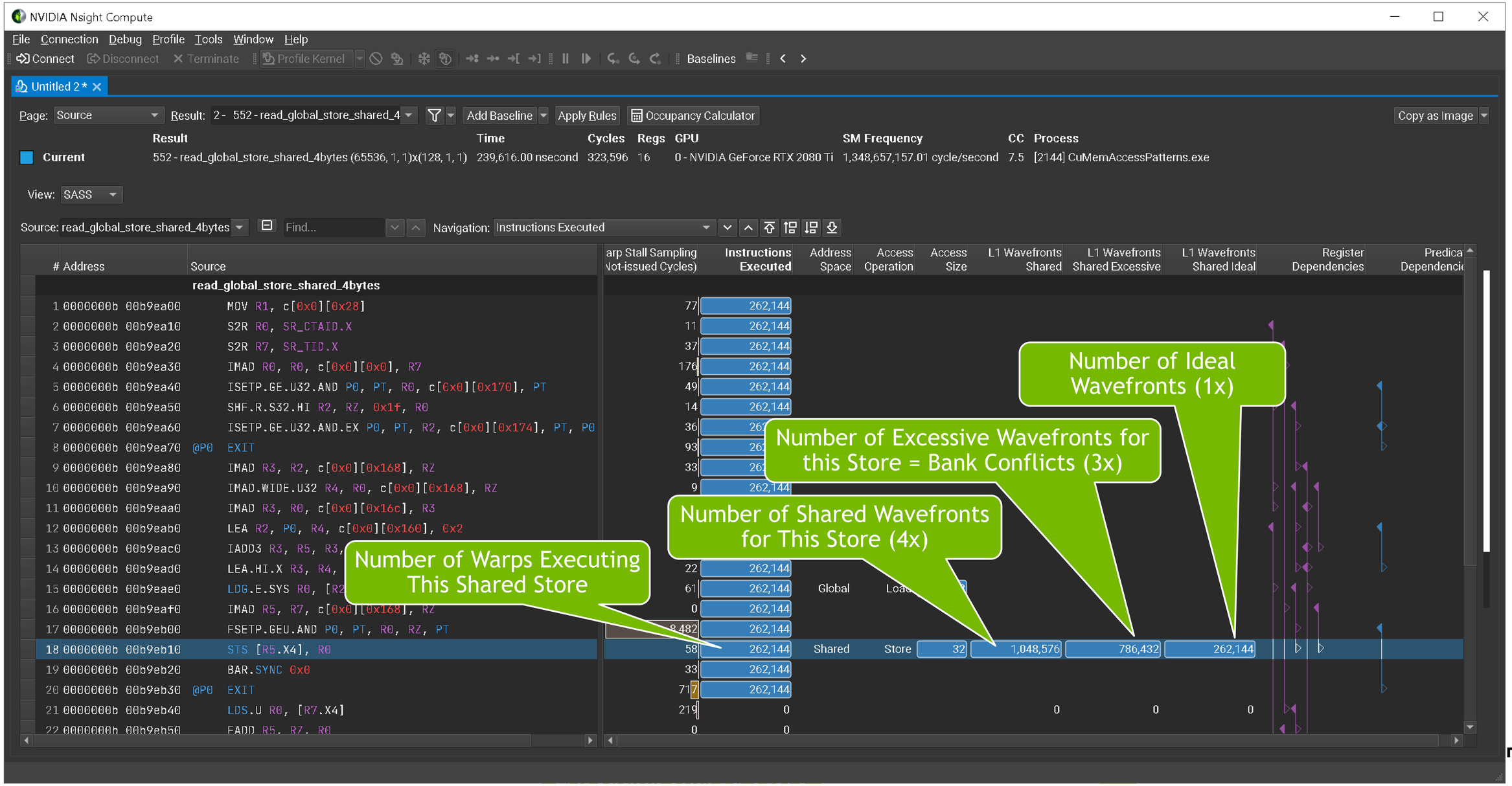


- Example Access
  - 1 warp, 32 threads
  - 4B per thread
  - Consecutive access with 16B stride
  - 3 Bank Conflicts
  - 4 Cycle to access

Also Referred to as Wavefronts Needed to Resolve the Request

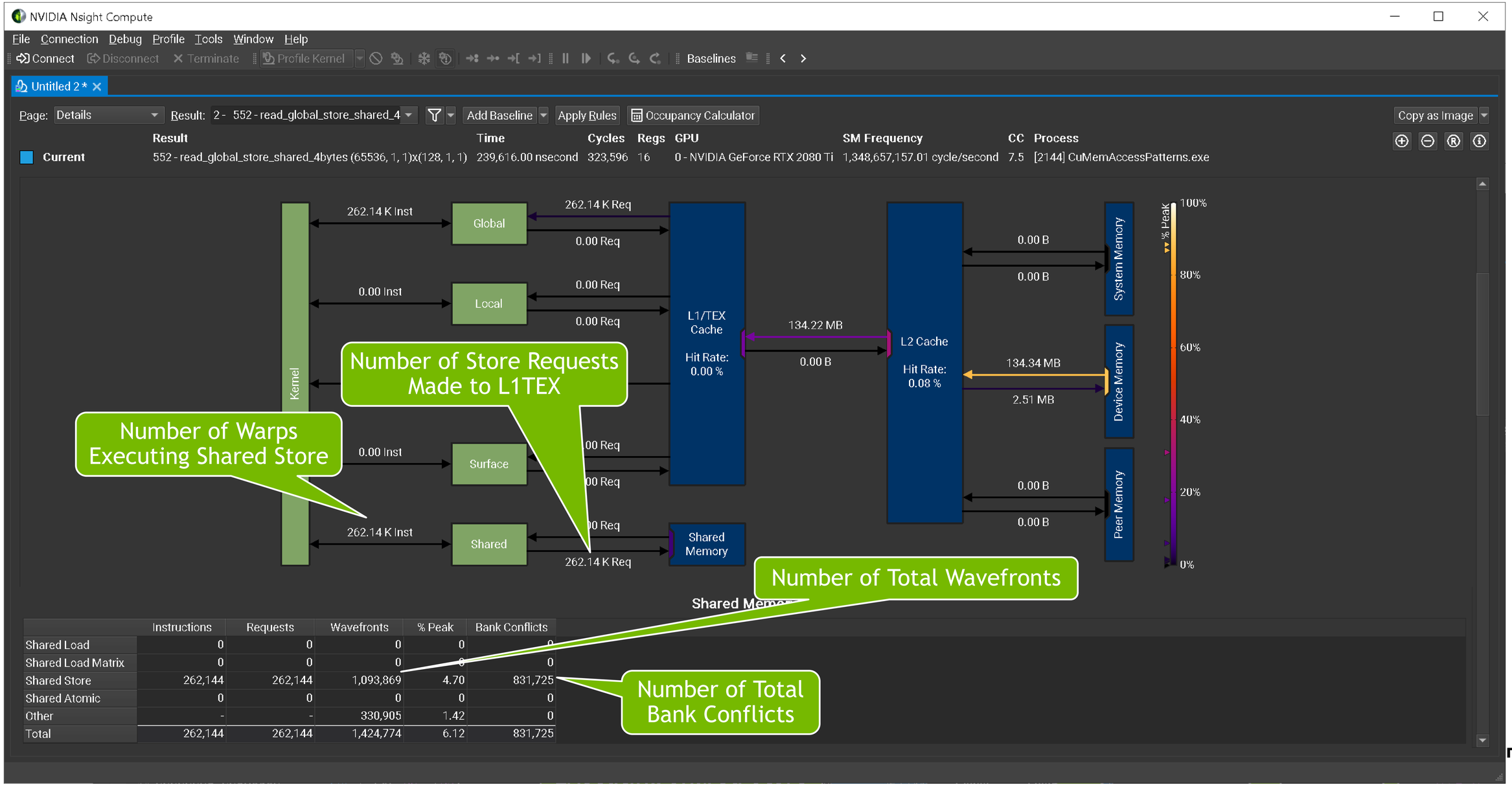
NSIGHT COMPUTE

Source Page



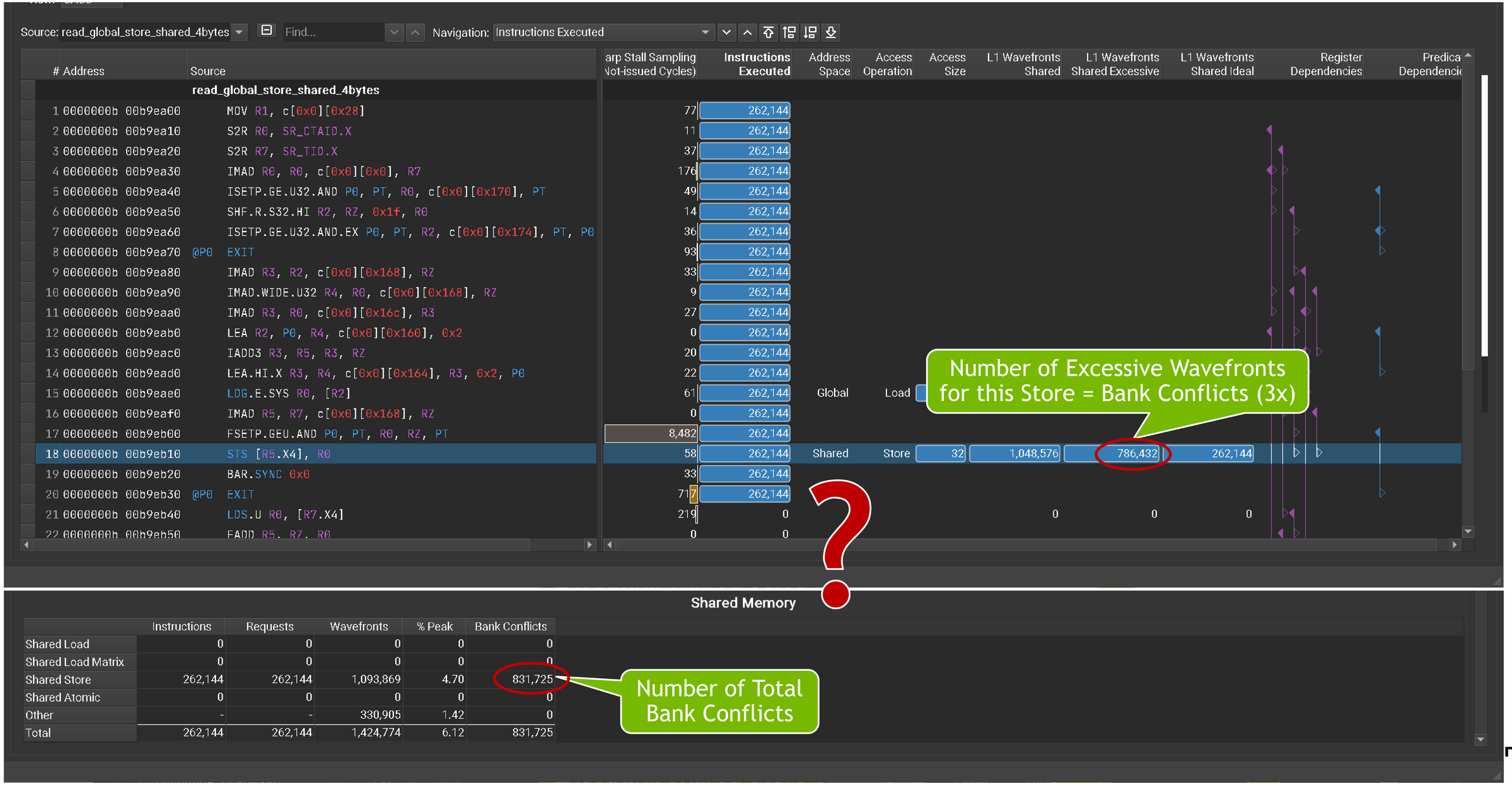
NSIGHT COMPUTE

Details Page



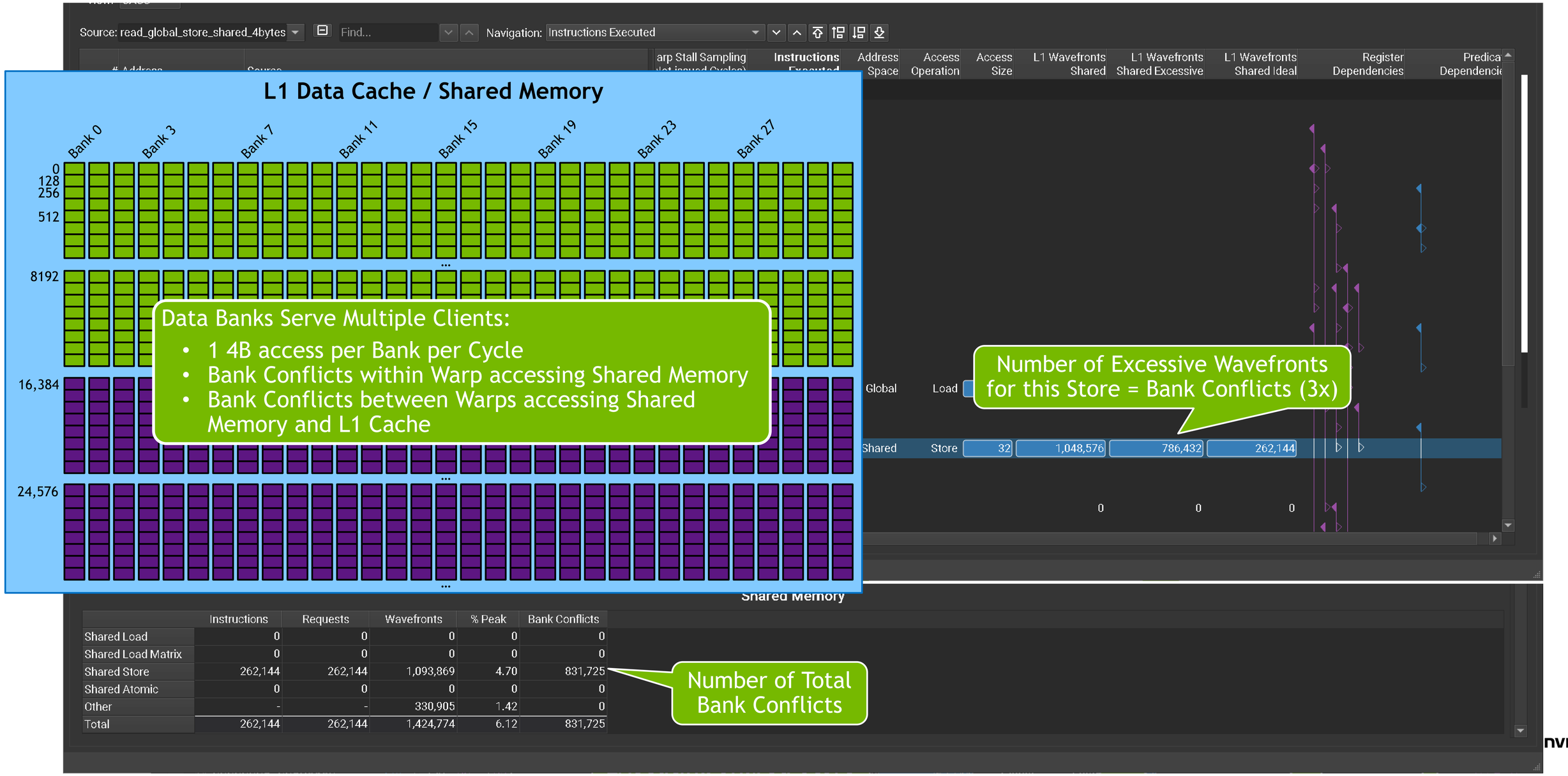
NSIGHT COMPUTE

Details Page



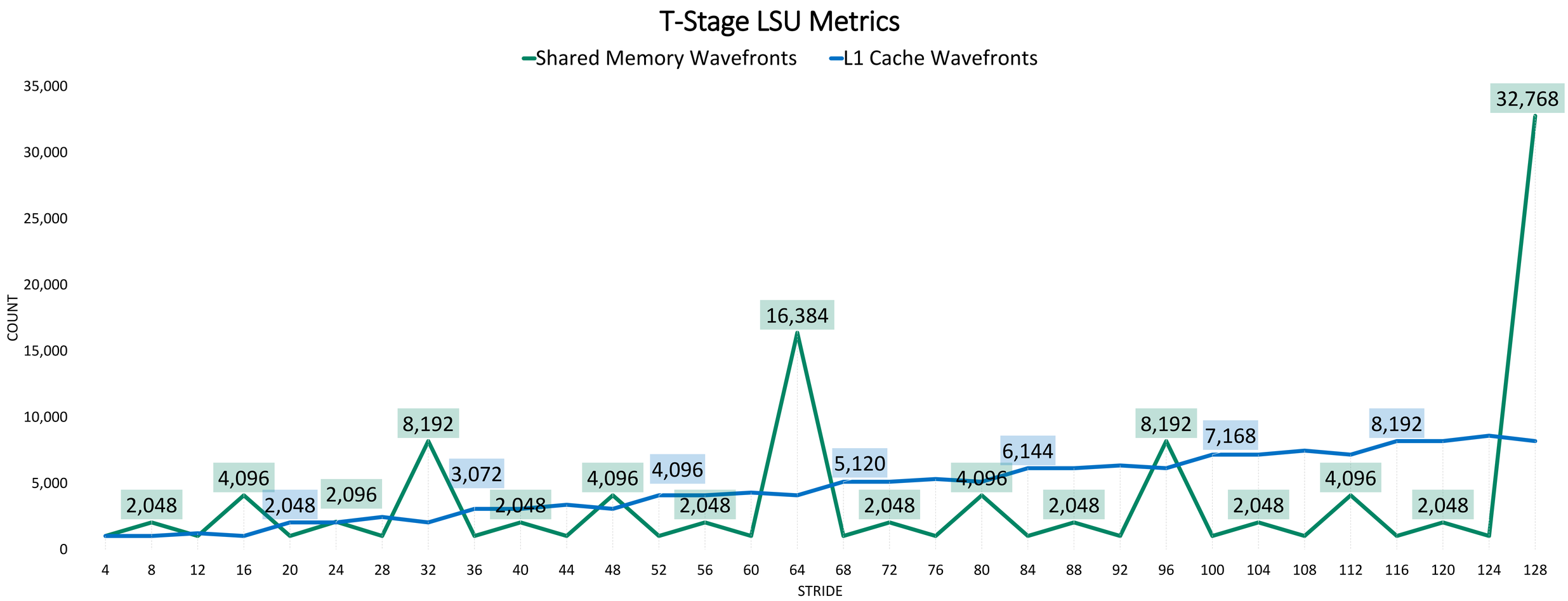
NSIGHT COMPUTE

Details Page



GLOBAL LOADS / SHARED ACCESSES

4-byte accesses & varying stride



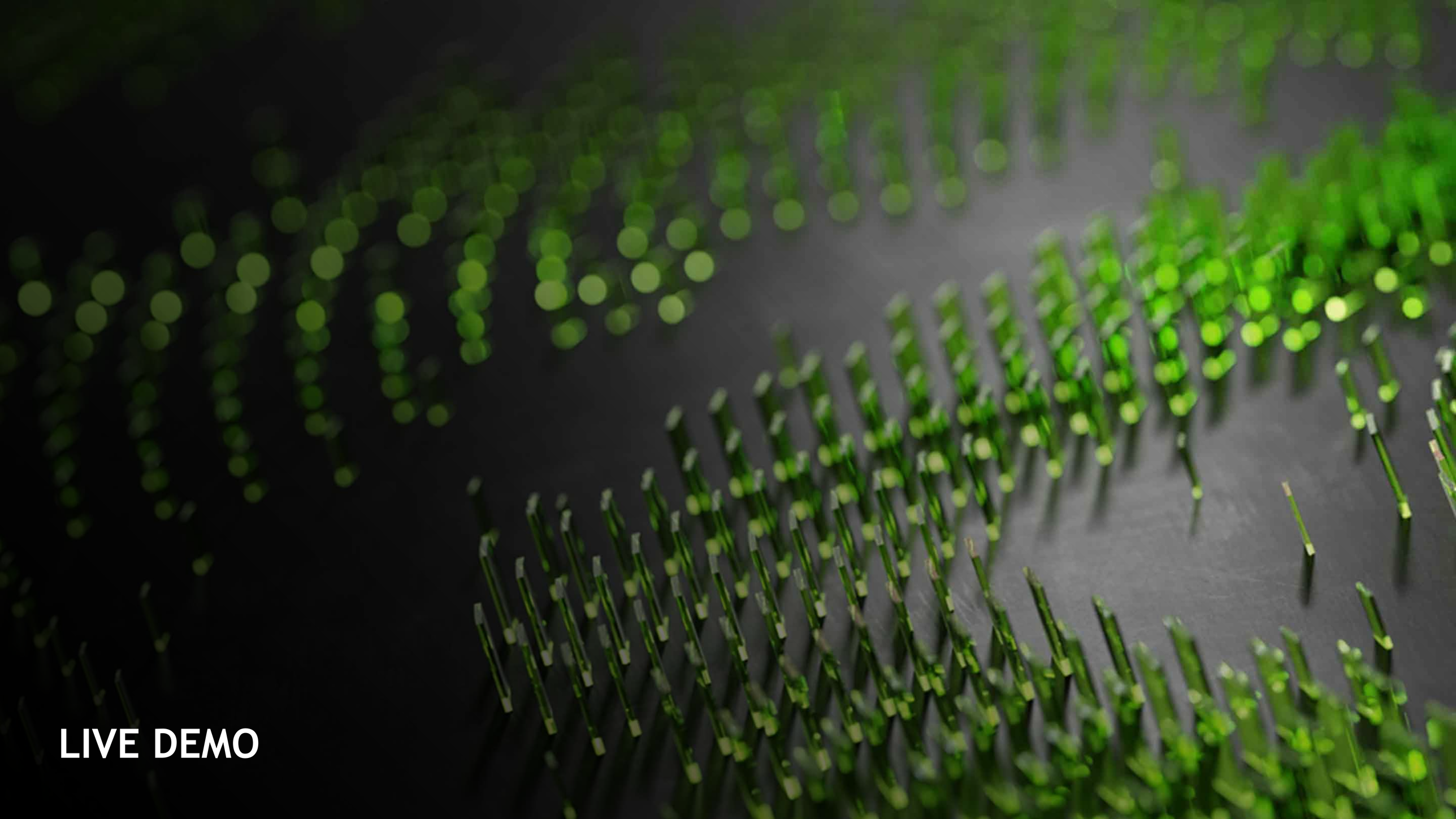
Shared Wavefronts: memory_l1_wavefronts_shared L1 **Wavefronts:** l1tex__t_output_wavefronts_pipe_lsu_mem_global_op_ld.sum L1 **Sectors:** l1tex__t_sectors_pipe_lsu_mem_global_op_ld.sum
Grid Size: 32,768 **# Warps:** 1,024

SHARED MEMORY

Takeaways

- Combined L1 Data Cache and Shared Memory
  - Common physical data memory
  - Common overall performance characteristics
  - Very different optimization strategies!

- Consider Shared Memory, when …
  - Accessing data multiple times
  - Using data synchronization across threads of a block
  - Cooperative parallel algorithms within threads of block
  - Benefiting from user-managed cache, i.e. you are in charge what is stored for how long
  - Requiring memory access pattern that do not fit well to L1 Cache Sectors or Cache Lines
    - ‘Diagonal’ accesses in memory banks
    - Sweet spots of strided accesses with no bank conflicts
  - Reducing or improving global memory traffic in L2
    - Use shared memory atomics for local reduction, then less global atomics
    - Shared atomics have higher throughput than global L2 atomics


LIVE DEMO

NSIGHT COMPUTE

Shipping Feature Highlights

- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies
- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules

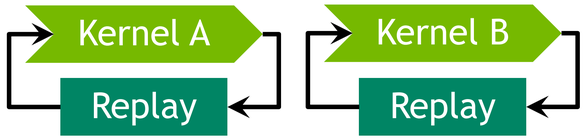
NSIGHT COMPUTE

Shipping Feature Highlights

- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies
- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules

Target Workloads:

Kernel/Application Replay:



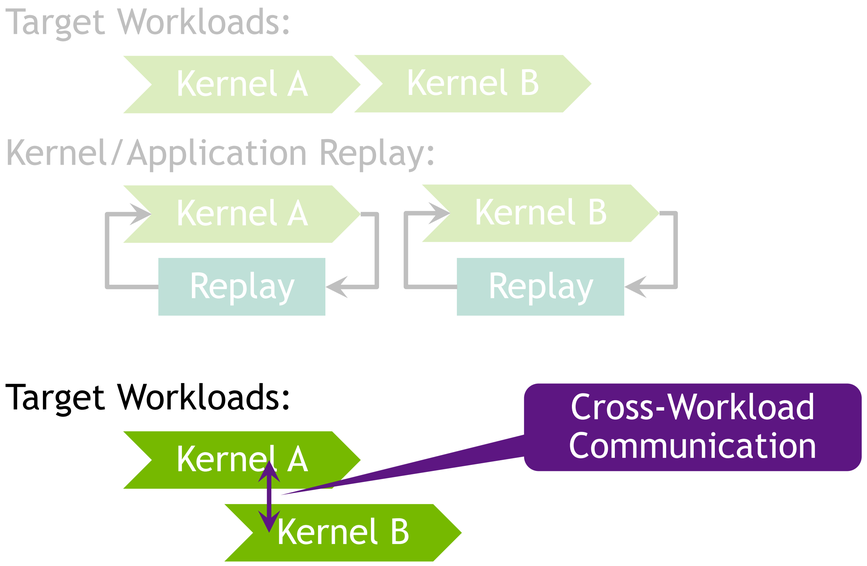
NSIGHT COMPUTE

Shipping Feature Highlights

- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies

- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page

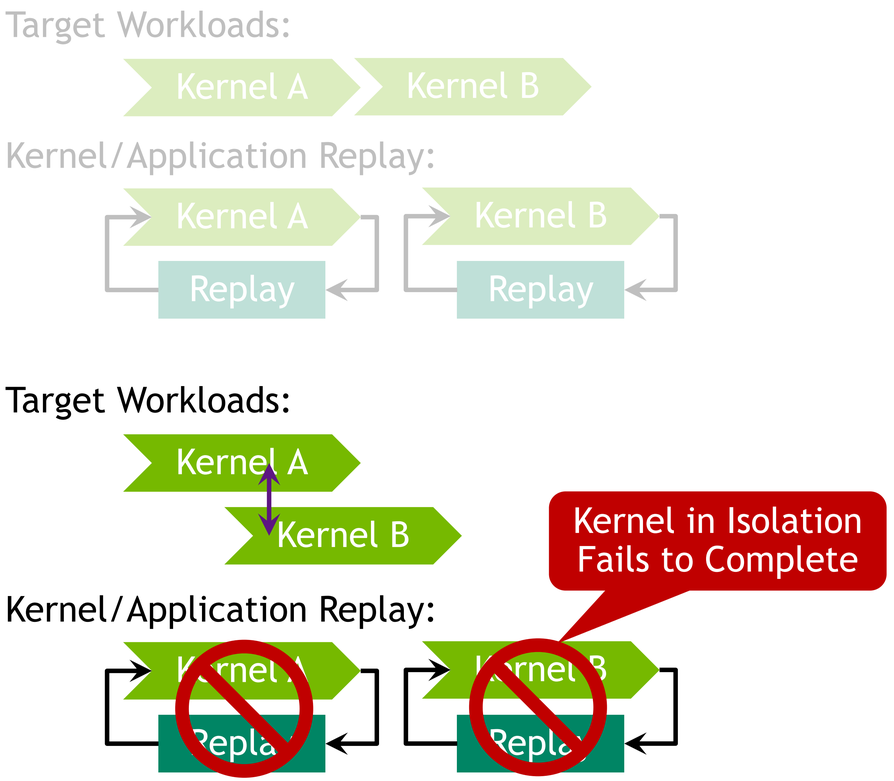
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules



NSIGHT COMPUTE

Shipping Feature Highlights

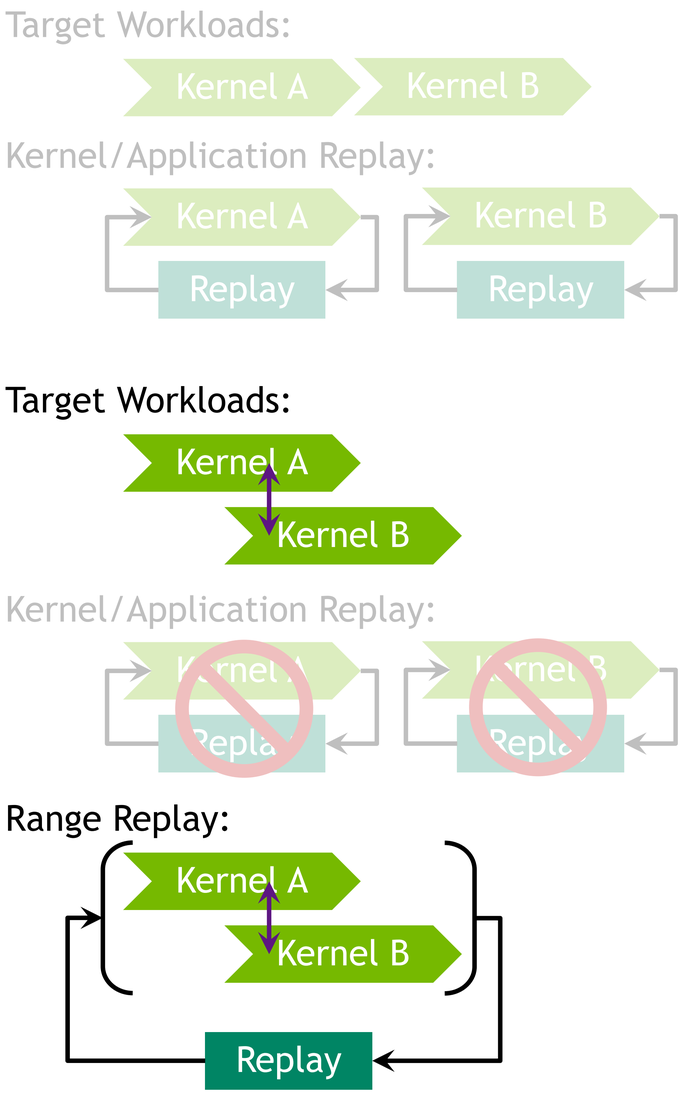
- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies
- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules



NSIGHT COMPUTE

Shipping Feature Highlights

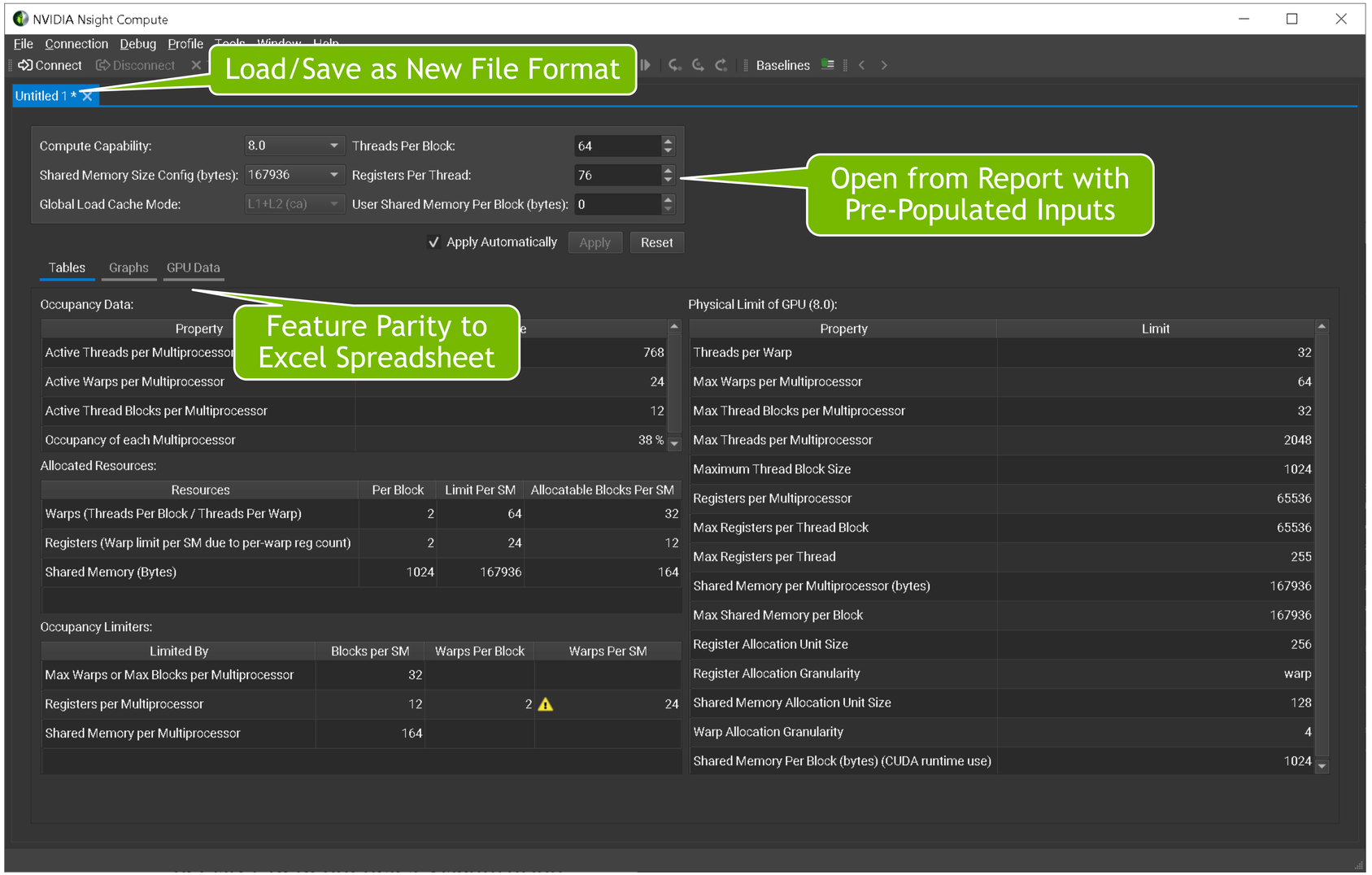
- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies
- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules



NSIGHT COMPUTE

Shipping Feature Highlights

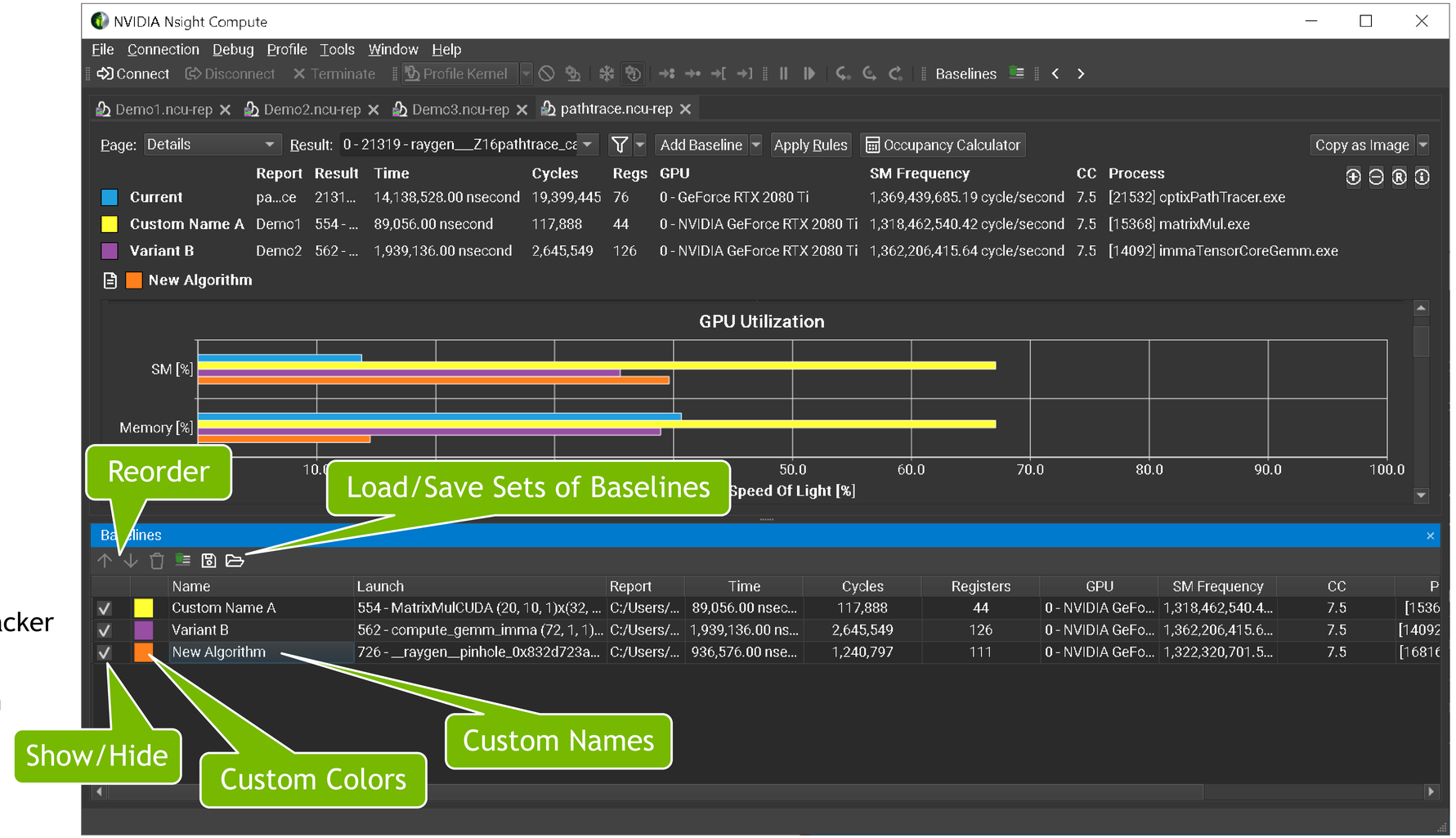
- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies
- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules



NSIGHT COMPUTE

Shipping Feature Highlights

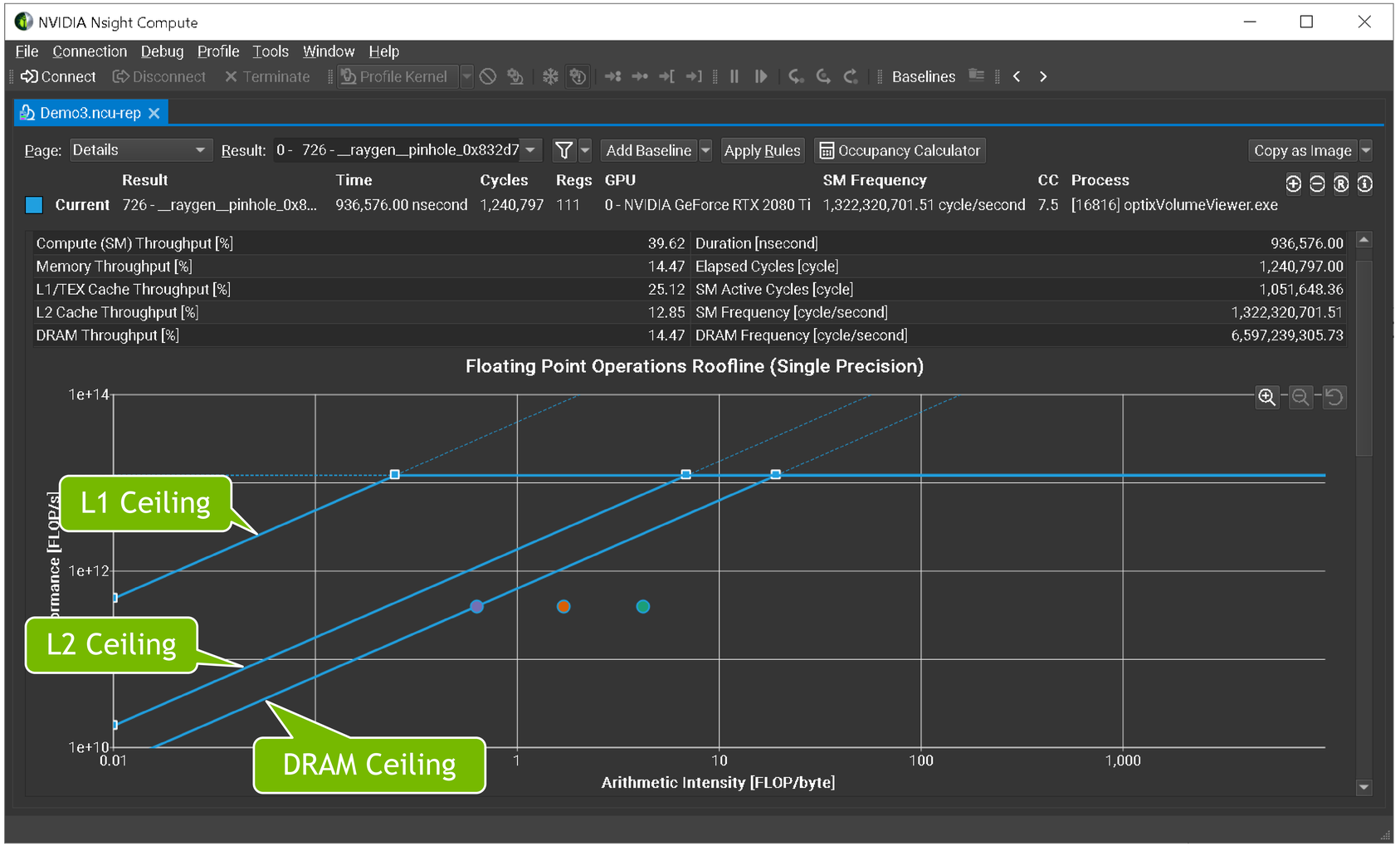
- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies
- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules



NSIGHT COMPUTE

Shipping Feature Highlights

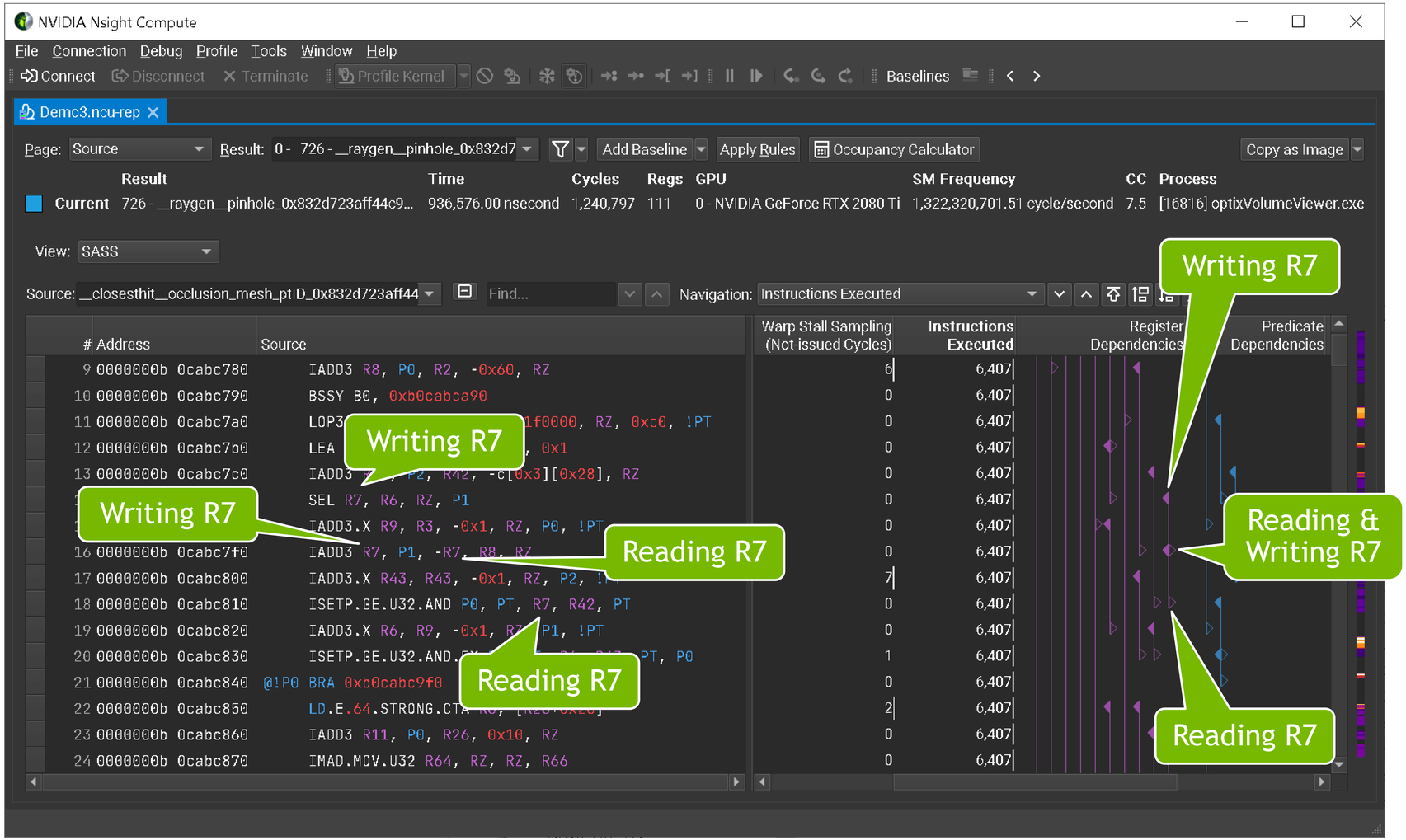
- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies
- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules



NSIGHT COMPUTE

Shipping Feature Highlights

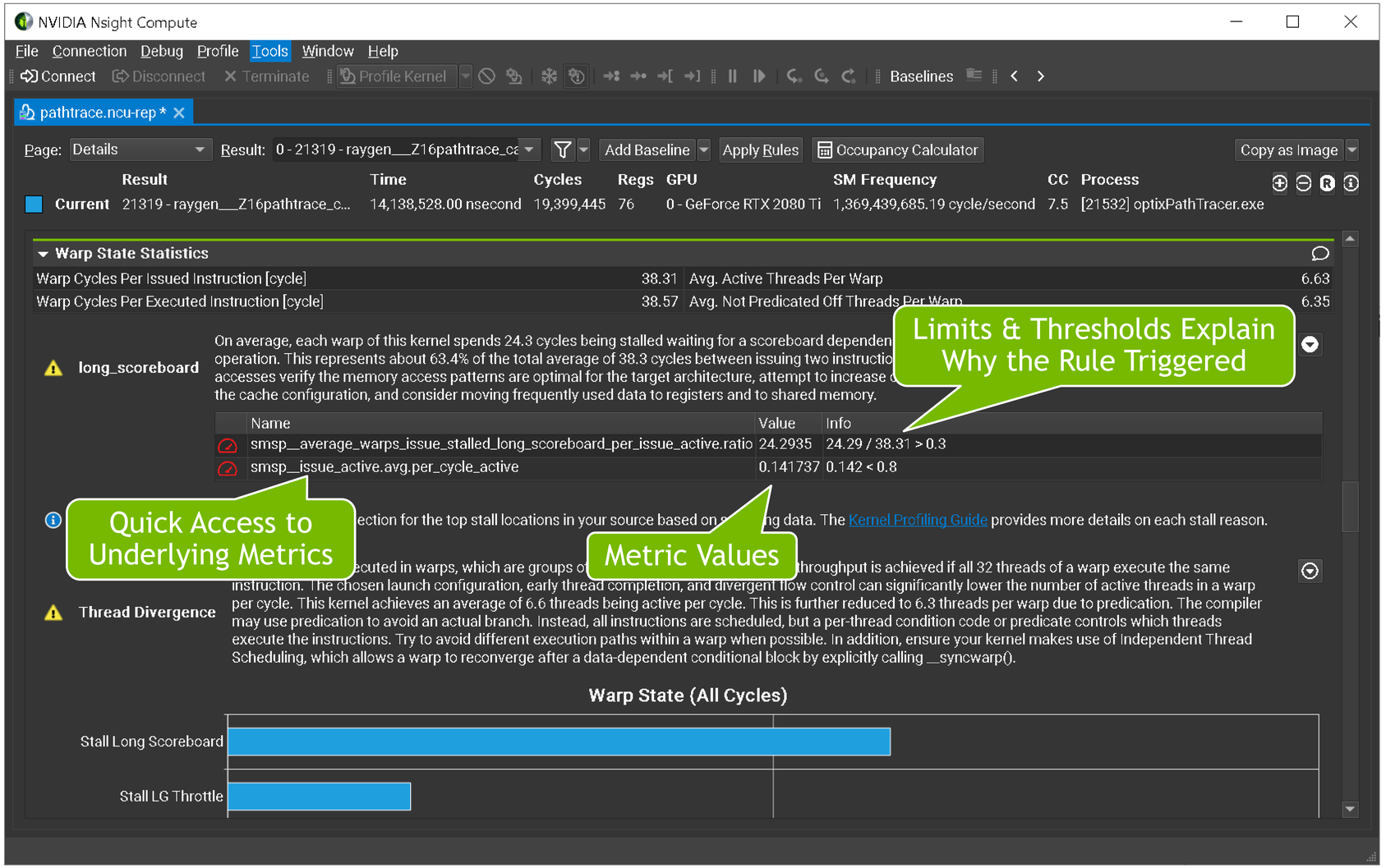
- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies
- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules



NSIGHT COMPUTE

Shipping Feature Highlights

- **2022.1 (CUDA 11.6):**
  - Range Replay
  - Metric Coverage for L2 ECC and Evict Policies
- **2021.3 (CUDA 11.5):**
  - Integrated Occupancy Calculator
  - Baselines Tool Window
  - Hierarchical Rooflines
  - CPU Call Stacks Capture
  - CLI Source Code Page
- **2021.2 (CUDA 11.4):**
  - Optix 7.x Support with Resource Tracker
  - Standalone CUBIN Source Viewer
  - Register Dependencies Visualization
  - Focus Metrics for Rules



NSIGHT COMPUTE

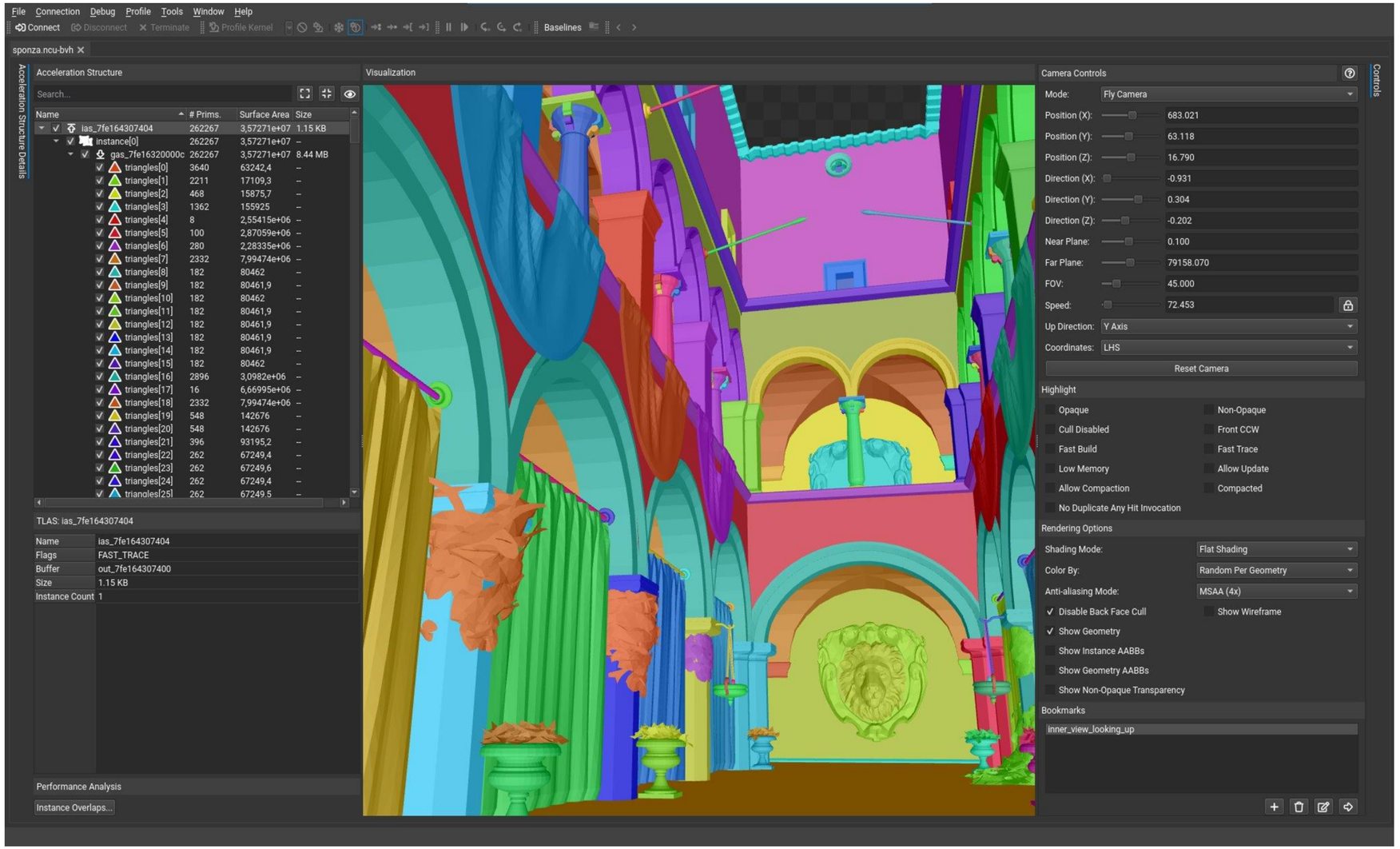
Upcoming Feature Highlights

- **2022.2**
  - Optix Support
    - Resource Tracker Updates
    - Acceleration Structure Viewer
  - Source Table
    - Metric Groups
    - Inline Function Table
  - Performance Improvements
  - Rules Updates

NSIGHT COMPUTE

Upcoming Feature Highlights

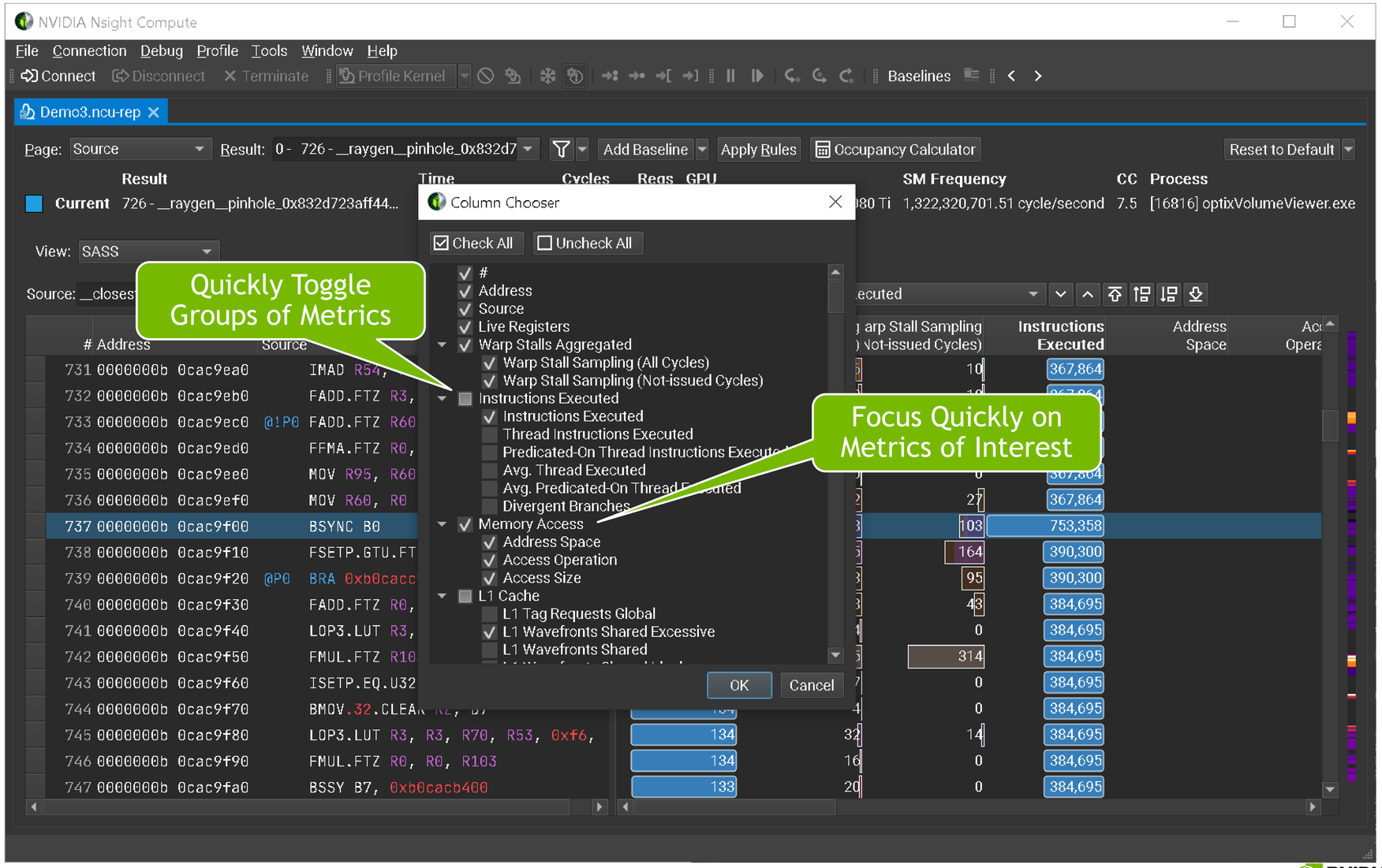
- **2022.2**
  - Optix Support
    - Resource Tracker Updates
    - Acceleration Structure Viewer
  - Source Table
    - Metric Groups
    - Inline Function Table
  - Performance Improvements
  - Rules Updates



NSIGHT COMPUTE

Upcoming Feature Highlights

- 2022.2
  - Optix Support
    - Resource Tracker Updates
    - Acceleration Structure Viewer
  - Source Table
    - Metric Groups
    - Inline Function Table
  - Performance Improvements
  - Rules Updates



FURTHER INFORMATION

For Nsight Compute

- **Download**
  https://developer.nvidia.com/nsight-compute
  Included in CUDA Toolkit installer, standalone installer available on web page

- **Documentation**
  https://docs.nvidia.com/nsight-compute

- **Feedback & Questions:**
  For direct feedback and bug reports from within the UI, use `Help > Send Feedback`
  Forum: https://devtalk.nvidia.com

- **Further Training:**
  https://developer.nvidia.com/nsight-compute-blogs
  https://developer.nvidia.com/nsight-compute-videos

DEVELOPER TOOLS ACROSS GTC

- **Sessions**
  - S41493 - What, Where, and Why? Use CUDA Developer Tools to Detect, Locate, and Explain Bugs and Bottlenecks
  - S41447 - Orin Developer Tools: The Next Frontier
  - S41500 - Optimizing Communication with Nsight Systems Network Profiling
  - S41518 - Killing Cloud Monsters Has Never Been Smoother
  - S41859 - GPU Performance Analysis and Improvements utilizing the NVIDIA Nsight Perf SDK

- **Labs**
  - DLIT2169 - Optimizing CUDA Machine Learning Codes with Nsight Profiling Tools
  - DLIT2207 - Debugging and Analyzing Correctness of CUDA Applications
  - DLIT2319 - Developer Tools Fundamentals for Ray Tracing using NVIDIA Nsight Graphics and NVIDIA Nsight Systems

- **Connect With Experts**
  - CWE41541 - What's in your CUDA toolbox: CUDA Profiling, Optimization, and Debugging Tools
  - CWE41887 - Connect with the Experts: Getting Started with Ray Tracing and NVIDIA's Ray Tracing Developer Tools

- **More Information at**
  - https://developer.nvidia.com/tools-overview

Thank you!