

# Maximum DirectStorage

Ani Alston

Graphics Engineer

04/2023

The Intel ARC logo is displayed in white text on a dark blue square background. The word "intel" is in a smaller, lowercase font, and "ARC" is in a larger, uppercase font. A small trademark symbol (TM) is located to the right of "ARC".

intel<sup>®</sup>  
ARC<sup>™</sup>

# Agenda

## Overview

GPU Decompression for Asset streaming


DirectStorage Architecture on Intel® Graphics

Writing Optimized DirectStorage 1.1 Application

Intel® Expanse & Microsoft BulkLoad Demo

Summary


# DirectStorage On Intel® GPUs



**DIRECTSTORAGE 1.0**

Reduced CPU Overhead	Faster Load times	Better Utilization of SSDs
Improved System Performance	PCIe 4.0 & 5.0 support	

CPU → GPU  
DECOMPRESSION



**DIRECTSTORAGE 1.1**

Reduced Load Times	Improved Throughput	GPU Decompression
High-Performance Assets Streaming	Software Advances with Modern Hardware	

# Agenda

Overview

GPU Decompression for Asset streaming

DirectStorage Architecture on Intel® Graphics

Writing Optimized DirectStorage 1.1 Application

Intel® Expanse & Microsoft BulkLoad Demo

Summary

# GPU Decompression Benefits

Faster level load times\*

Free CPU cycles

Reduce system bandwidth utilization

## Requirements

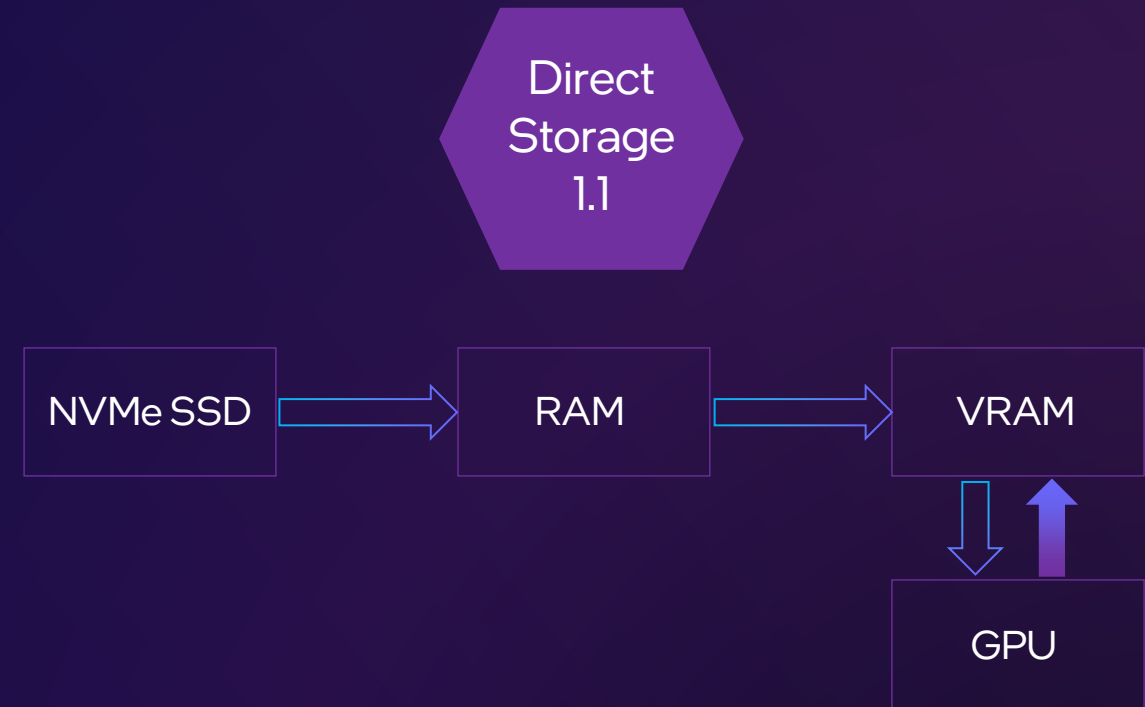
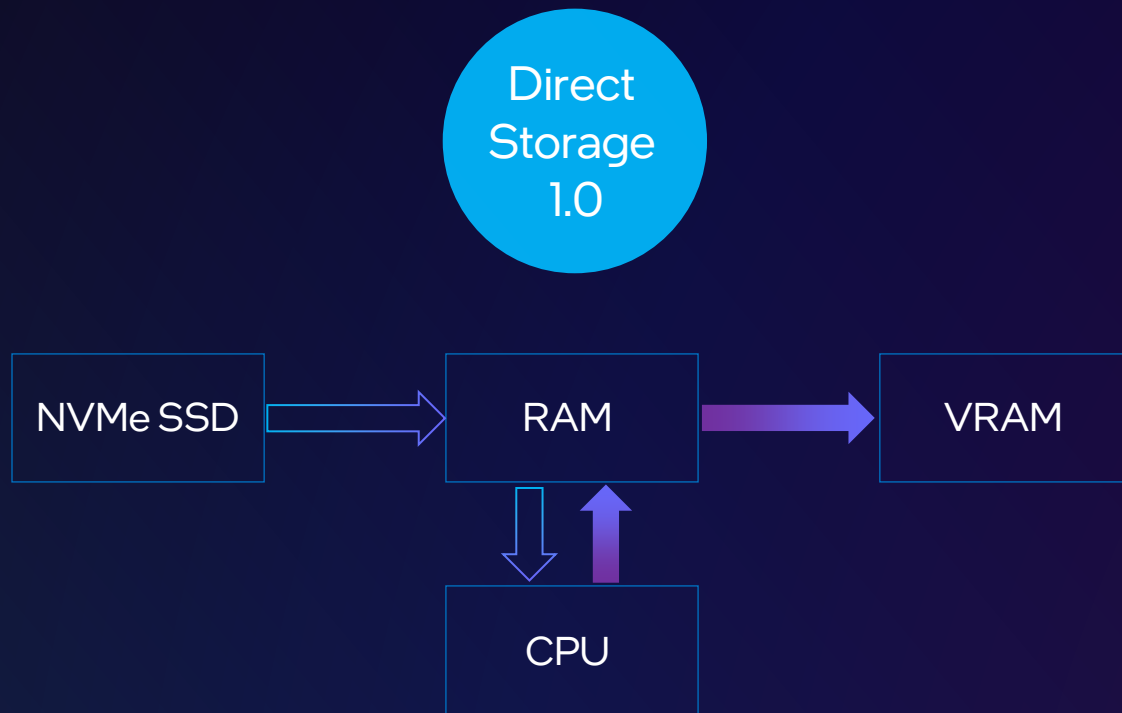
- DX12 Ultimate
- GPU with Shader Model 6.0 support
- NVMe SSD (recommended)
- Windows 11 (recommended)



A scene from Expanse showing nearly 1,000 textures, each over 350 MB in size, uncompressed, using about 100 MB of physical GPU memory.

*\*Pre-production feature, results may vary based on system configuration*

# Game asset streaming optimization



Key:

- Compressed Data
- Decompressed Data



# Agenda



Overview

GPU Decompression for Asset streaming

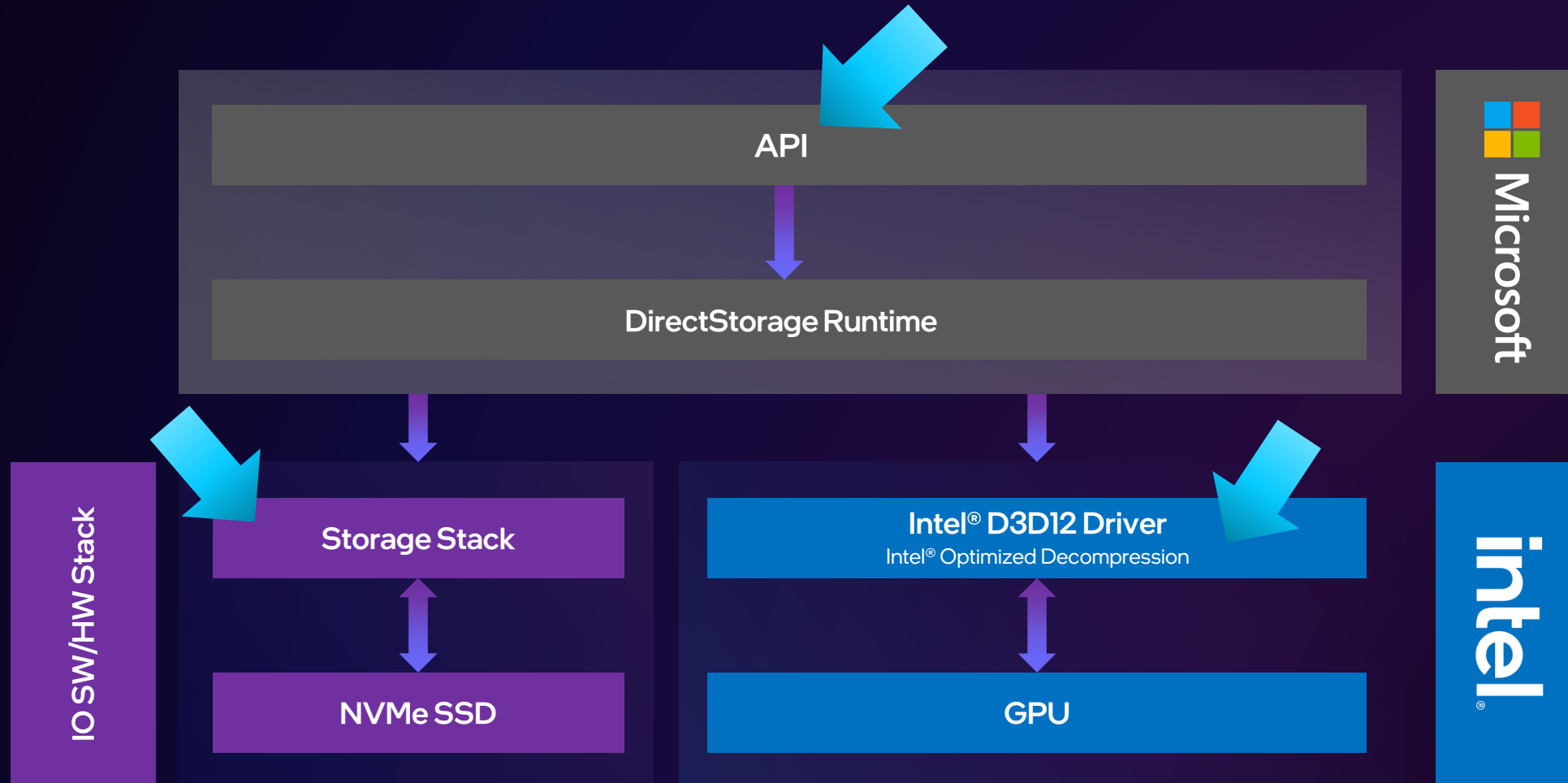
DirectStorage Architecture on Intel® Graphics

Writing Optimized DirectStorage 1.1 Application

Intel® Expanse & Microsoft BulkLoad Demo

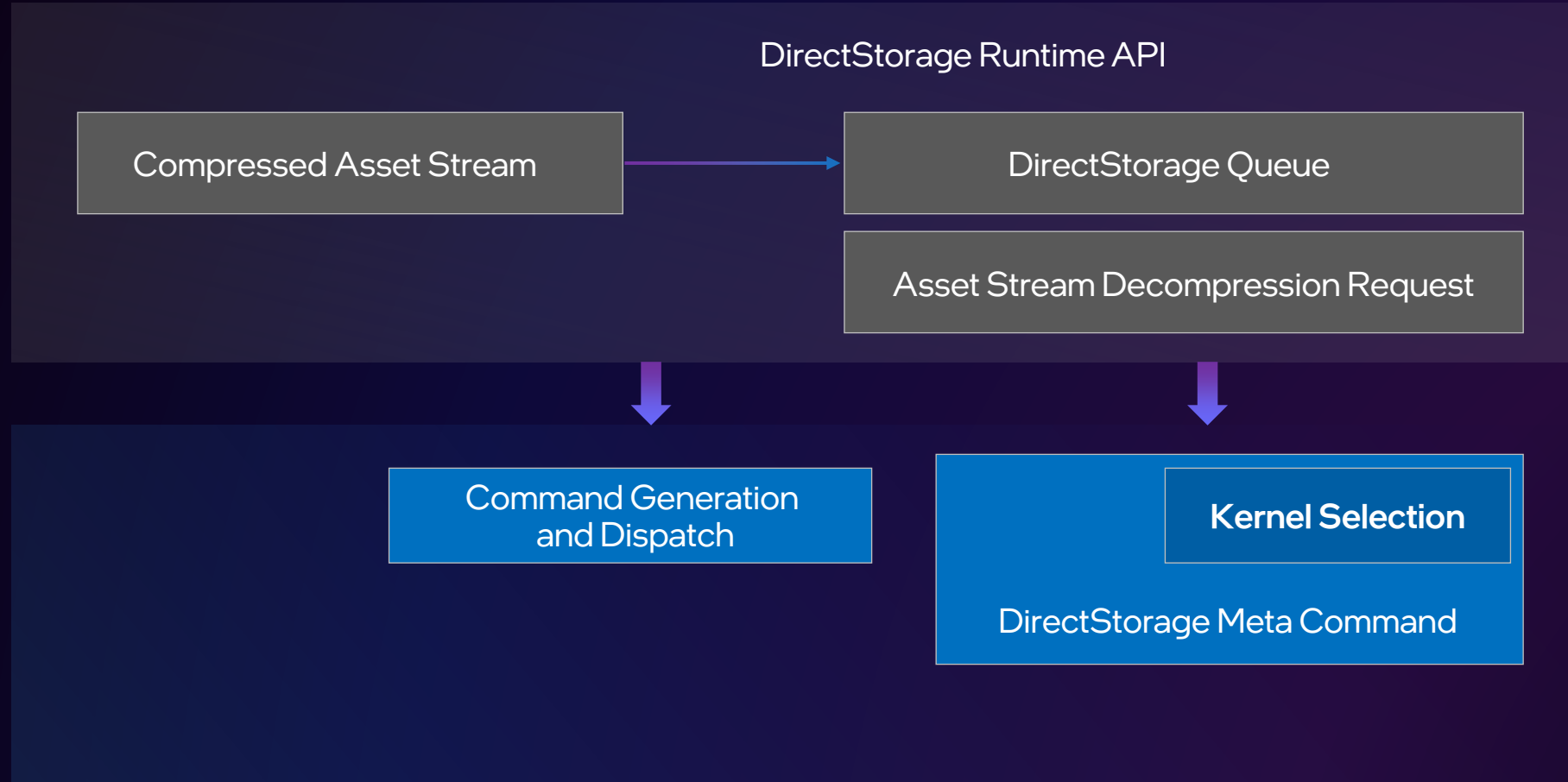
Summary

# DirectStorage Stack





# Driver Resident Acceleration



# Agenda

Overview

GPU Decompression for Asset streaming

DirectStorage Architecture on Intel® Graphics

Writing Optimized DirectStorage 1.1 Application

Intel® Expanse & Microsoft BulkLoad Demo

Summary

# Build Iconic Scenes

1

Real-time texture streaming, demo updated with DirectStorage 1.1 (GPU HW Decompression)

2

Built on top of the Sampler Feedback and Virtual Texture\* tech, although DirectStorage doesn't depend on it

3

Loading from:  
**350GB** disk assets (16k x 16k textures)

4

Using only:  
**128MB** Staging buffer + **~230MB** Runtime texture space of VRAM



\* Virtual textures can add overhead if not managed correctly

<https://github.com/GameTechDev/SamplerFeedbackStreaming>

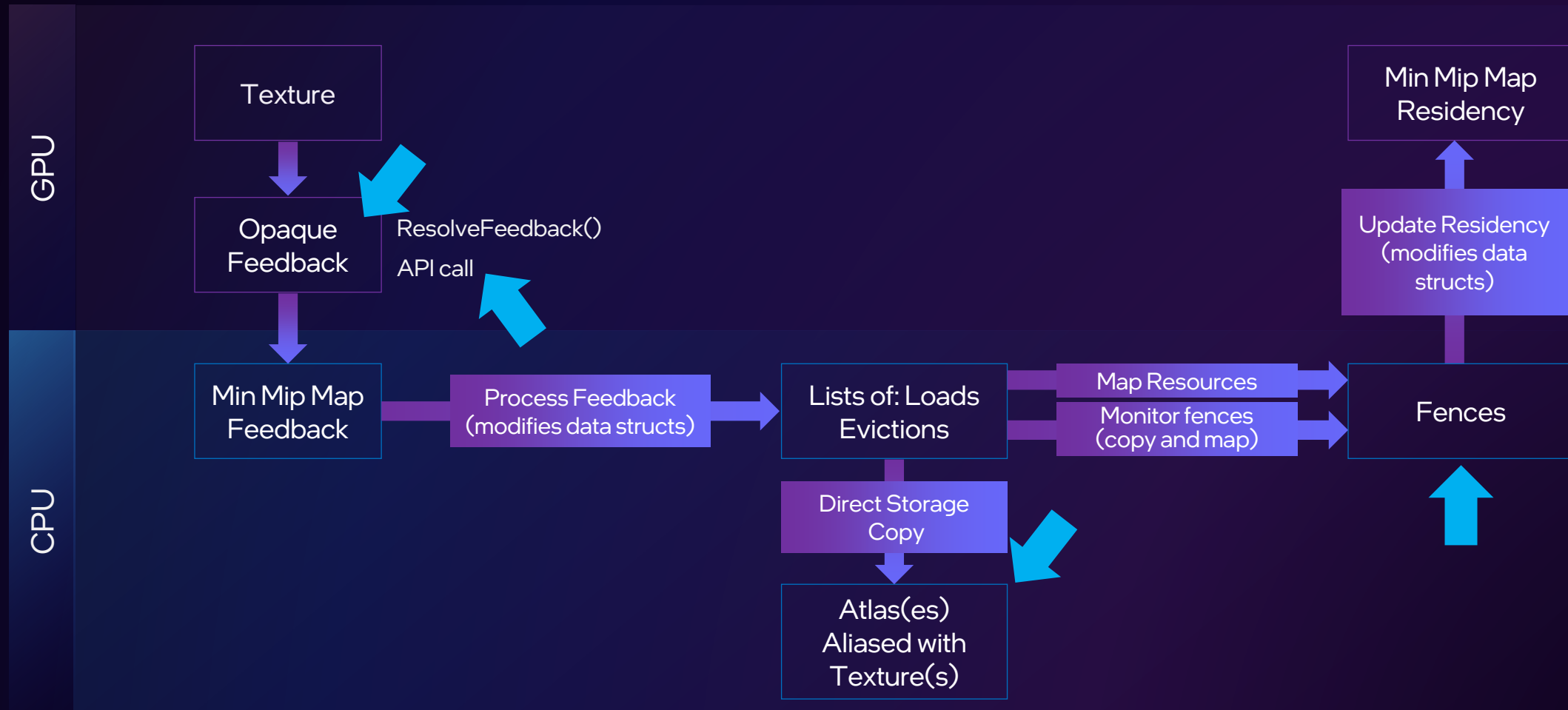
# Debug view of the mip sampler feedback



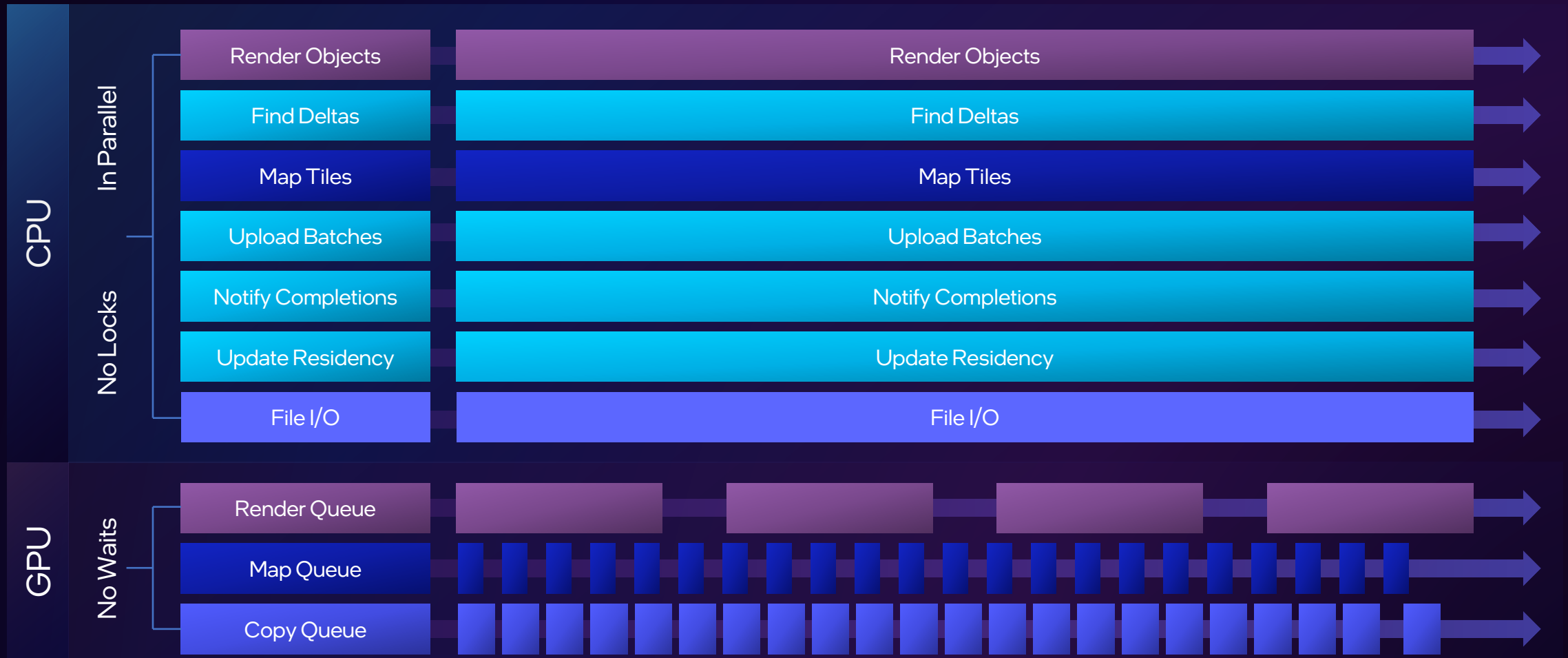


# Sample(sampler, clamp)

WriteSamplerFeedback() in HLSL



# Async Queues





# DirectStorage Code Example: Creating Objects

```
ComPtr<IDStorageFactory> dsFactory;  
ComPtr<IDStorageFactory> dsFile;  
ComPtr<IDStorageQueue> dsQueue;  
  
HRESULT h1 = DStorageGetFactory(IID_PPV_ARGS(&dsFactory));  
HRESULT h2 = dsFactory->OpenFile(in_path, IID_PPV_ARGS(&dsFile));  
  
DSTORAGE_QUEUE_DESC queueDesc{};  
queueDesc.Capacity = DSTORAGE_MAX_QUEUE_CAPACITY;  
queueDesc.Priority = DSTORAGE_PRIORITY_NORMAL;  
queueDesc.SourceType = DSTORAGE_REQUEST_SOURCE_FILE;  
queueDesc.Device = pD3D12Device;  
HRESULT h3 = dsFactory->CreateQueue(&queueDesc, IID_PPV_ARGS(&dsQueue));
```

<https://learn.microsoft.com/en-us/windows/win32/dstorage/dstorage-interfaces>

# DirectStorage Code Example: Loading a Texture Tile

```
DSTORAGE_REQUEST request{};
request.Options.CompressionFormat = DSTORAGE_COMPRESSION_FORMAT_GDEFLATE;
request.Options.SourceType = DSTORAGE_REQUEST_SOURCE_FILE; // 1 Bit member, only disk or mem
request.Options.DestinationType = DSTORAGE_REQUEST_DESTINATION_TILES; // mem, buffer, tex region, mips

request.Source.File.Source = in_dsFileHandle;
request.Source.File.Offset = fileOffset;
request.Source.File.Size = numBytes; // 64KB or less if compresses

request.Destination.Tiles.Resource = pD3DResource;
request.Destination.Tiles.TiledRegionStartCoordinate = D3D12_TILED_RESOURCE_COORDINATE{x, y, 0, mip};
request.Destination.Tiles.TileRegionSize = D3D12_TILE_REGION_SIZE{1, FALSE, 0, 0, 0};
request.UncompressedSize = D3D12_TILED_RESOURCE_TILE_SIZE_IN_BYTES;

dsQueue->EnqueueRequest(&request);
```

<https://learn.microsoft.com/en-us/windows/win32/dstorage/dstorage-enumerations>

Can be blocking

# DirectStorage Code Example: Loading a Texture Tile

```
dsQueue>EnqueueSignal(fence.Get(), fenceValue++);  
dsQueue>Submit();
```



Signal before submit



DirectStorage will auto-submit when its queues fill

# Agenda

A person wearing a headset is shown in profile, looking at a computer monitor in a dimly lit room. The person is wearing a white t-shirt and a headset with a microphone. The background is dark with some blue light from the monitor.

Overview

GPU Decompression for Asset streaming

DirectStorage Architecture on Intel® Graphics

Writing Optimized DirectStorage 1.1 Application

Intel® Expanse & Microsoft BulkLoad Demo

Summary



# Expanse demo

Intel(R) Arc(TM) A770 Graphics

0.400	Spin
0.400	Camera

Roller Coaster

Bandwidth (MB/s) avg = 28.171

GPU ms: Feedback | Draw  
4.82 | 2.761

CPU ms: Feedback | Draw | Frame  
0.42 | 0.50 | 8.76

Reserved KB: 218175808  
Committed KB: 97856 (0.04 %)  
Heap Occupancy KB: 6.22% of 1572864

985 Num Objects

Terrain Object Feedback Viewer

Misc. Options

Texture Visualize

Tile Min Mip Overlay

DEMO MODE

BENCHMARK MODE

VIDEO PREVIEW

Expanse live demo stats  
expanse.exe -maxnumobjects  
985 -numospheres 9999 -  
hidefeedback -camerate 0.4 -  
animationRate 0.4 -  
lightFromView

# Result from Expanse

## stress.bat

- timingstart 200
- timingstop 700
- capturetrace traceplayer.exe
- file uploadTraceFile\_1.json
- mediadir media
- staging 128

- File bytes to read (per iter): 25,469,019,672
- Number of requests: 407879
- Staging buffer size: 128 MB
- # iterations: 4
- Bandwidth: 4971.87 MB/s from disk
- Bandwidth: 5218.17 MB/s uncompressed to GPU

\*Performance may vary.

Test system - Graphics: Intel® Arc™ A770 16GB Graphics, Graphics Driver: 4257, Processor: Intel® Core™ i9-12900K, MSI MPG Z690, BIOS: 1.10, Memory: 32GB (2x16GB) DDR5 @ 4800MHz, Storage: Samsung 980 Pro NVMe, OS: Windows 11 Version 22H2



# Optimization findings from Expanse: Factors Affecting Bandwidth



Mapping Time (UpdateTileMappings)



Pipelining – Staging buffer size



Request Size and Number of  
Requests in Flight

# UpdateTileMappings

Tip

Performance can degrade over time due to heap fragmentation creating a bottleneck in the pipeline when moving data from CPU to GPU



# Pipelining: Staging Buffer Size

Tip

The right value for **SetStagingBufferSize** is rather important when decompression is enabled



Chose a value based on profiling as it helps improve pipelining between SSD loads and GPU decompression



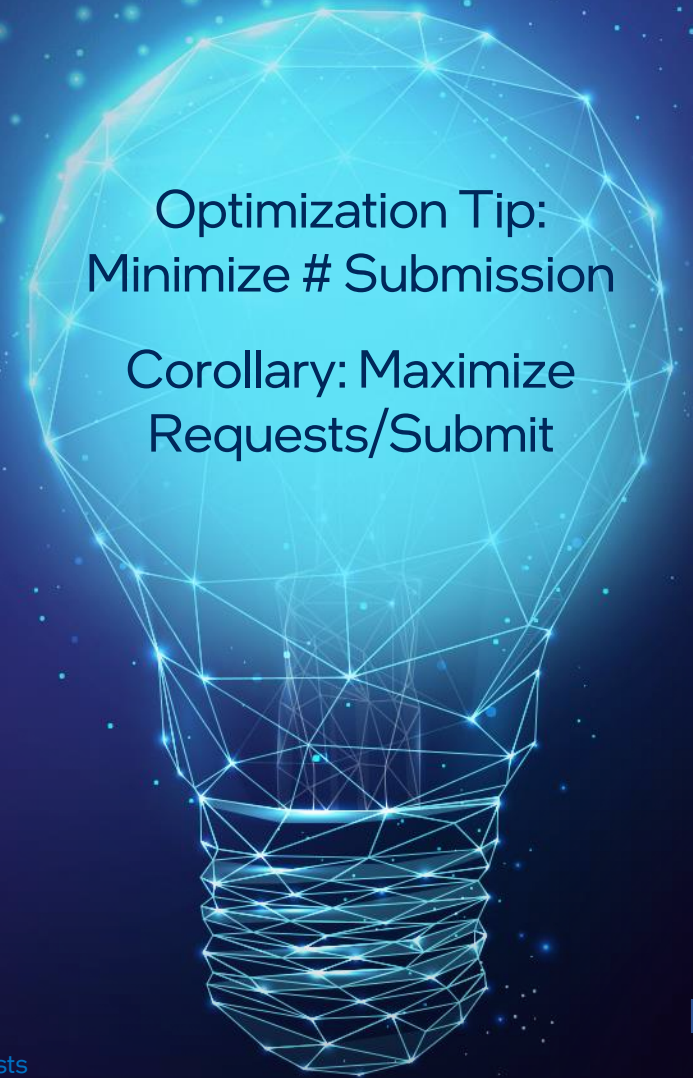
# Request size and Requests in Flight

Tip

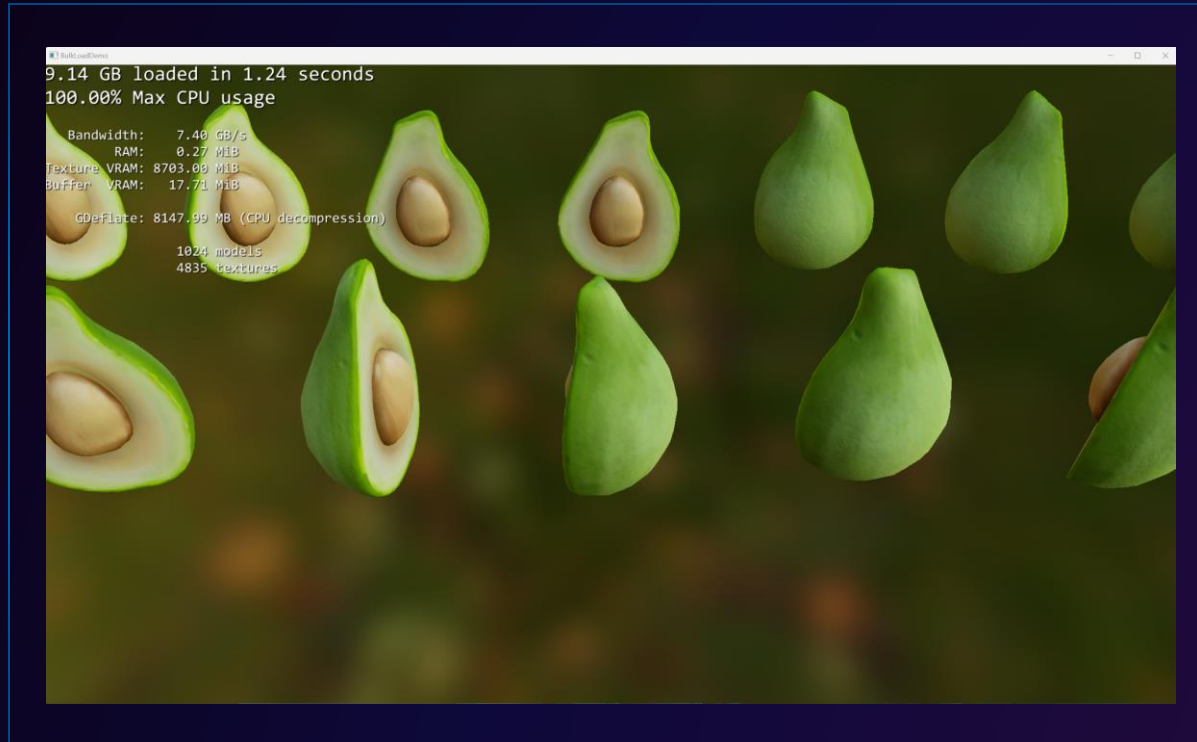
How does latency  
in SSD's and  
GPU's work?



Optimization Tip:  
Minimize # Submission  
Corollary: Maximize  
Requests/Submit



# MSFT BulkLoad demo

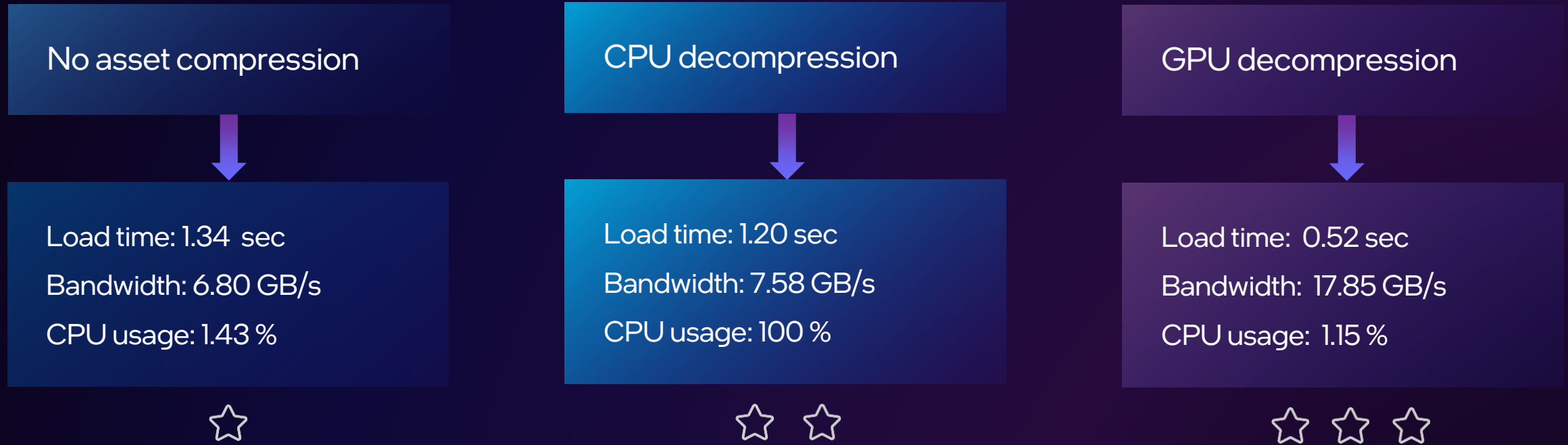


Uncompressed asset  
No runtime decompression

Compressed Asset  
CPU decompression

Compressed asset  
GPU decompression

# BulkLoad demo runs results (9.14 GB)



\*Performance may vary.

Test system - Graphics: Intel® Arc™ A770 16GB Graphics, Graphics Driver: 4257, Processor: Intel® Core™ i9-12900K, MSI MPG Z690, BIOS: 1.10, Memory: 32GB (2x16GB) DDR5 @ 4800MHz, Storage: Samsung 980 Pro NVMe, OS: Windows 11 Version 22H2



# Demos Takeaways

- Games can eliminate extended load times
- Streaming made faster and easier to implement
- Unleash developers' imagination with instant access to hundreds of gigabytes of data



```
Load Iteration Number : 1
9.14 GB loaded in 0.48 seconds
1.27% Max CPU usage
Bandwidth: 19.09 GB/s
RAM: 0.27 MiB
Texture VRAM: 8703.00 MiB
Buffer VRAM: 17.71 MiB
GPU late: 8147.99 ms (GPU decompression)
```



\*Compressed images, not representative of actual demo

# Agenda



Overview

GPU Decompression for Asset streaming

DirectStorage Architecture on Intel® Graphics

Writing Optimized DirectStorage 1.1 Application

Intel® Expanse & Microsoft BulkLoad Demo

Summary

# Summary

Intel Iris X<sup>e</sup> Graphics and Intel Arc GPUs support DirectStorage 1.1

Begin developing with DirectStorage 1.1 today

We can't wait to see how innovative developers will use this feature!  
Email [gamedevtech@intel.com](mailto:gamedevtech@intel.com) for questions

# References

Resource	URL
Expanse Demo - Sampler Feedback Streaming With DirectStorage	<a href="https://github.com/GameTechDev/SamplerFeedbackStreaming">https://github.com/GameTechDev/SamplerFeedbackStreaming</a>
Bulk Load Demo	<a href="https://github.com/microsoft/DirectStorage/tree/main/Samples/BulkLoadDemo">https://github.com/microsoft/DirectStorage/tree/main/Samples/BulkLoadDemo</a>
DirectStorage 1.1 Now Available	<a href="https://devblogs.microsoft.com/directx/directstorage-1-1-now-available/">https://devblogs.microsoft.com/directx/directstorage-1-1-now-available/</a>
DirectStorage 1.1 for Intel GPUs	<a href="https://www.intel.com/content/www/us/en/developer/articles/news/directstorage-on-intel-gpus.html">https://www.intel.com/content/www/us/en/developer/articles/news/directstorage-on-intel-gpus.html</a>
DirectStorage API reference	<a href="https://learn.microsoft.com/en-us/windows/win32/dstorage/dstorage-api-reference">https://learn.microsoft.com/en-us/windows/win32/dstorage/dstorage-api-reference</a>
DirectStorage enumerations	<a href="https://learn.microsoft.com/en-us/windows/win32/dstorage/dstorage-enumerations">https://learn.microsoft.com/en-us/windows/win32/dstorage/dstorage-enumerations</a>



# Acknowledgements

Hisham Chowdhury

Sreenivas Kothandaraman

Daniele Pieroni

Alexander Kharlamov

Marissa Du Bois

Ethan Davis

Daniel Jacobsen

Ashley Gregory

Vinod Tipparaju

Pradeep Radhakrishna

Patrick Farrell

Pete Brubaker

Allen Hux



Damyan Pepper

Cassie Hoef

Cooper Partin



Thank you



# Notices & Disclaimers

Performance varies by use, configuration and other factors. Learn more at [www.intel.com/PerformanceIndex](https://www.intel.com/PerformanceIndex) (graphics and accelerators).

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. No product or component can be absolutely secure.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

Intel technologies may require enabled hardware, software or service activation.

All product plans and roadmaps are subject to change without notice.

Code names are used by Intel to identify products, technologies, or services that are in development and not publicly available. These are not "commercial" names and not intended to function as trademarks.

Statements that refer to future plans or expectations are forward-looking statements. These statements are based on current expectations and involve many risks and uncertainties that could cause actual results to differ materially from those expressed or implied in such statements. For more information on the factors that could cause actual results to differ materially, see our most recent earnings release and SEC filings at [www.intc.com](https://www.intc.com).

© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.

intel®