Vulkan Ray Tracing is Here!

Set of provisional extension specifications are publicly available today for industry feedback

First beta developer drivers are shipping today

Coherent ray tracing framework seamlessly integrated into existing Vulkan functionality - flexible merging of rasterization and ray tracing

Familiar to users of existing proprietary ray tracing APIs but also introduces new implementation flexibility

Hardware agnostic - can be accelerated on existing GPU compute and dedicated ray tracing cores

Primary focus on meeting desktop market demand for both real-time and offline rendering

The industry’s first open, cross-vendor, cross-platform standard for ray tracing acceleration
Ray Tracing Refresher

Rasterization and ray tracing can both use triangles to describe scene geometry - but only ray tracing calculates physical phenomena such as shadows, reflections and refractions.

Ray tracing calculates how rays intersect and interact with scene geometry, materials and light sources.


Ray Tracing is a Flexible Technique

Programmers need programmable flexibility to trace rays through scenes for a wide variety of visual effects - some examples...

Ambient Occlusion

Reflections

Shadows

Global Illumination
Step 1: Create Efficient Scene Geometry

Ray tracing may use a huge numbers of rays
Specialized data structures for interrogating scene
geometry are necessary for efficient acceleration

Acceleration Structures
Contains low-level 3D geometry to be ray traced and
high-level references into the geometry
Opaque internal organization details
Each vendor can optimize for processing for their hardware
E.g. Bounding Volume Hierarchy (BVH) for rapidly determining
if there is any geometry in the path of a ray

Build Acceleration Structure
Vulkan driver integrates supplied geometry
into its two-level Acceleration Structure

Using a BVH data structure to enable
efficient ray tracing through a 3D scene
Step 2: Traverse Scene with Rays

Two ways to traverse Acceleration Structure
Launching rays into scene to generate results

Ray Tracing Pipelines
A new type of graphics pipeline
Implicit management of ray intersections
Application compiles a set of shaders into the pipeline to provide desired ray and material processing

Ray Queries
Any type of shader can launch a ray at any time
Shader can process intersection data however it wishes
Shader controls how traversal proceeds

Model courtesy of PTC
Traversing with Ray Tracing Pipelines

Implicit Ray and Shader Execution Management

Application compiles collection of shaders to be invoked on ray/geometry intersection into the Ray Tracing Pipeline.

Controlling which shaders are invoked during traversal enables a wide variety of ray tracing techniques.

Hit Shaders can query the materials they intersect e.g. transparent materials can be handled differently than opaque.

Intersection and Hit shaders can control how traversal proceeds.

Shader stages can communicate parameters and results through ray payload structures.

1. Launch 2D/3D grid of rays into scene contained in an Acceleration Structure.

2. ‘Intersection’ Shader computes ray intersections. Ray-triangle intersections are built-in.

3. Invoke ‘Any Hit’ Shader if intersection is found. Multiple intersections possible - arbitrary order.

4. Invoke ‘Closest Hit’ shader on the closest intersection of the ray OR invoke ‘Miss’ Shader if no hit is found. Can trace more rays.

Ray Tracing Pipeline
Traversing with Ray Queries

Explicit Ray Management within a Single Shader

Any standard shader (e.g. compute, vertex and fragment shaders) can invoke a single ray traversal at any time.

Uses an Acceleration Structure and a geometric description of the ray being traced.

Shader reads intersection properties during traversal and controls how materials are processed and how the traversal proceeds.

1. Shader launches a single ray into scene contained in an Acceleration Structure.
2. Shader takes action depending on intersection properties. Can trace more rays.

Graphics, Compute or Ray Tracing Pipeline
Vulkan Ray Tracing includes GLSL and SPIR-V Extensions

Enabling compiled GLSL/SPIR-V shaders to operate in a Ray Tracing Pipeline - similar to HLSL features used in Direct3D’s DXR

HLSL and Vulkan with DXC
Microsoft’s DXC HLSL compiler was open sourced in Jan 2017
Google and others have added SPIR-V code generation to DXC with Microsoft’s knowledge and approval
Vulkan developers can now choose between GLSL and HLSL!

HLSL for Vulkan Ray Tracing
NVIDIA added code generation to DXC to generate SPIR-V for the NVIDIA vendor ray tracing extensions from HLSL
Vulkan Ray Tracing Extensions supported in HLSL soon

Developers that wish to run their ray tracing applications on both Vulkan Ray Tracing and DXR will be able to port their HLSL shaders with minimal changes
Pipeline Libraries

Ray Tracing Pipelines can use many shaders
Potentially orders of magnitude more shaders (1000s) than traditional applications to handle diverse tracing techniques and material types

Compilation Bottleneck
Compiling many shaders into a Ray Tracing Pipeline can be computationally intensive and cause application bottlenecks and stuttering

Vulkan Pipeline Library Extension
Enables a library of SPIR-V shaders to be incrementally compiled into an existing Ray Tracing Pipeline saving significant processing load

Multiple shaders used to build complex lighting in a Quake 2 scene
Host Offload of Setup Operations

Ray tracing setup compute workloads can be intensive
Building Acceleration Structures and compiling Ray Tracing Pipelines

Two Vulkan mechanisms to offload and control setup workloads
on the host CPU(s) for smoother, faster rendering

Build Acceleration Structure on Host
Use the host to build Acceleration Structure in host memory and then copy to the GPU - rather than build directly on the GPU
Final size of Application Structure is known before copying to the GPU - enabling optimized GPU memory allocation

Deferred Host Operations
Driver returns deferred work handle to application for later execution
Application controls work execution and can chose to distribute onto multiple cores and background threads

Deferred Host Operations can be used together to asynchronously use multiple CPU cores to build Acceleration Structures on the host

Using Deferred Host Operations to build a complex Acceleration Structure using multiple CPU cores to offload the work from the GPU for faster, smoother framerates
Vulkan Ray Tracing and DXR

Similar Acceleration Structure / Ray Tracing Shader architectures enables straightforward porting of raytracing functionality between Vulkan Raytracing and DXR including re-use of ray tracing shaders written in HLSL.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Vulkan Ray Tracing</th>
<th>DX12 / DXR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ray Tracing Pipelines</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Ray Queries</td>
<td>Optional</td>
<td>DXR 1.1</td>
</tr>
<tr>
<td>Language for Ray Tracing Shaders</td>
<td>GLSL or HLSL</td>
<td>HLSL</td>
</tr>
<tr>
<td>Pipeline Libraries</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Build Acceleration Structure on Host</td>
<td>Optional</td>
<td>No</td>
</tr>
<tr>
<td>Deferred Host Operations</td>
<td>Optional</td>
<td>No</td>
</tr>
<tr>
<td>Capture/Replay Support for Tools</td>
<td>Optional</td>
<td>No</td>
</tr>
</tbody>
</table>

(e.g. RenderDoc)
Call for Industry Feedback

- Khronos seeking developer feedback before finalizing Vulkan ray tracing specs
  - Including what functionality should be mandated
  - GitHub Release Tracker, GitHub Feedback Issue or through the Khronos Developer Slack

- Additional Resources
  - Specifications available on the Vulkan Registry
  - Blog with many more details
  - Press Release

Real time Vulkan Ray Tracing effects in Wolfenstein: Youngblood
Background

March 2020
Khronos Connects Software to Silicon

Open interoperability standards to enable software to effectively harness the power of multiprocessors and accelerator silicon

3D graphics, XR, parallel programming, vision acceleration and machine learning

Non-profit, member-driven standards-defining industry consortium

Open to any interested company

All Khronos standards are royalty-free

Well-defined IP Framework protects participant’s intellectual property

>150 Members ~ 40% US, 30% Europe, 30% Asia

This work is licensed under a Creative Commons Attribution 4.0 International License
Khronos Active Initiatives

3D Graphics
- Desktop, Mobile, Web
- Embedded and Safety Critical

3D Assets
- Authoring
- and Delivery

Portable XR
- Augmented and
- Virtual Reality

Parallel Computation
- Vision, Inferencing,
- Machine Learning

Guidelines for creating APIs to streamline system safety certification
Khronos Exploratory Group Process

Enables Khronos and the wider industry, to inclusively build consensus on whether to undertake a new standardization initiative - without committing any company to IP licensing obligations.

- Khronos members and/or non-members
- Idea for a new Khronos Initiative
  - Khronos organizes a discussion forum (Open to all)
  - Initiative Proposal
    - Khronos establishes an Exploratory Group to CREATE a Statement of Work (Under NDA, Open to all)
    - SOW
      - No IP Commitments
        - High-level requirements and use case discussions only
      - Detailed Cooperative Designs
        - Work and participants protected by Khronos IP Framework

Khronos Boards votes to approve progression to each stage.

Khronos Members
ANARI Working Group

ANARI - Analytic Rendering Interface API
Analytic Rendering is image generation performed primarily to gain and communicate insights into complex data sets primarily for scientific visualization and data analytics.

SITUATION BEFORE
Visualization Apps and Engines must be ported to multiple APIs

SITUATION AFTER
Cross-vendor API to provide access to state-of-the-art rendering across multiple platforms
ANARI Architecture and Support

Rather than specifying the details of the rendering process, ANARI will enable a visualization application to simply describe the relationship between objects in a scene to be rendered and leave the details of the rendering process to a backend renderer.
Get Involved!

- Information on Khronos Standards: www.khronos.org
- Any company is welcome to join Khronos: https://www.khronos.org/members/

Benefits of Khronos membership

- Gain early insights into industry trends and directions
- Influence the design and direction of key open standards that will drive your business
- Accelerate your time-to-market with early access to specification drafts
- Gather industry requirements for future open standards
- Draft Specifications Confidential to Khronos members
- Publicly Release Specifications and Conformance Tests
- Network with domain experts from diverse companies in your industry
- State-of-the-art IP Framework protects your Intellectual Property
- Enhance your company reputation as an industry leader through Khronos participation