# KHR SOUS GROUP

# The Vulkan Sessions

## The Vulkan Working Group (and friends) SIGGRAPH - July 31, 2019

## The Vulkan Sessions (part I)

2:00	<ul> <li>Vulkan Update</li> <li>Tom Olson (Arm), Piers Daniell (NVIDIA), Yuriy O'Donnell (Epic Games)</li> <li>Vulkan Timeline Semaphores</li> <li>James Jones (NVIDIA)</li> <li>OctaneRender on Vulkan</li> <li>Jules Urbach (OTOY)</li> <li>DXC Update: HLSL to SPIR-V for Vulkan</li> <li>Ehsan Nasiri (Google)</li> </ul>
3:40	Break

## The Vulkan Sessions (part II)



## Vulkan Update

Tom Olson, Arm Chair, Vulkan Working Group

Piers Daniell, NVIDIA Member, Machine Learning TSG Chair, OpenGL / OpenGL ES WG

Yuriy O'Donnell, Epic Games Member, Ray Tracing TSG



## Vulkan Update - Outline

**Platform News** 

**New Applications** 

**Working Group Activities** 

## Task Sub Group (TSG) Updates



## Platform news: Stadia

#### Google's game streaming platform

- Announced at GDC 2019
- Will go live in November 2019
- Console-in-the-cloud
- Vulkan 1.1 is standard

HR

 $\mathbf{\mathbf{\Sigma}}$ 





GDC 2019: The line to get into the keynote...

#### Stadia support in Vulkan

• VK\_EXT\_pipeline\_creation\_feedback

## Platform news: Android Q

#### Introducing Android Q Beta

13 March 2019

Posted by Dave Burke, VP of Engineering

https://android-developers.googleblog.com/2019/03/introducing-android-q-beta.html

# =1293008

#### Vulkan everywhere

We're continuing to expand the impact of Vulkan on Android, our implementation of the low-overhead, cross-platform API for high-performance 3D graphics. Our goal is to make Vulkan on Android a broadly supported and consistent developer API for graphics. We're working together with our device manufacturer partners to make Vulkan 1.1 a requirement on all 64-bit devices running Android Q and higher, and a

## Platform news: Vulkan on MacOS / iOS

#### **Vulkan Portability Initiative**

- Expose a Vulkan subset on closed platforms
- Shim libraries
- Shader translation tools
- SDK and validation



**Current focus** 

- Vulkan conformance test







## Lots of new games!



S S N S N S N

K H R

## Looking back to SIGGRAPH 2018...



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

© The Khronos® Group Inc. 2019 - Page 10

## Looking back to SIGGRAPH 2018...



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

© The Khronos<sup>®</sup> Group Inc. 2019 - Page 11

## SIGGRAPH 2019: It's a product!

#### **Adobe Premiere Rush**

Т

 $\mathbf{\mathbf{\Sigma}}$ 

- A pro-quality, cross-platform video capture and editing system

#### Vulkan powered on Android

- Supported on select Samsung, Google, and OnePlus phones
- Several hundred thousand lines of OpenCL C compute shaders
- Translated to SPIR-V by clspv
- Run in Vulkan command buffers

#### A testament to the power of SPIR-V!



## CAD on Vulkan?

People are starting to ask the question...

#### CAD on Vulkan makes sense

- CAD apps make lots of draw calls
- CAD apps are often CPU bound

#### But CAD needs things Vulkan doesn't have

- E.g., stippled lines
- VK\_EXT\_line\_rasterization shipped this week
- Other CAD support features under consideration



## **Emulating other APIs on Vulkan**

#### Vulkan as Hardware Abstraction Layer (HAL)

- A very thin layer over the hardware
- A good target for middleware
- E.g., game engines!

#### Using Vulkan as a HAL to support other APIs

- ANGLE: OpenGL ES
- Zink: OpenGL
- DXVK: D3D 10/11
- D9vk: D3D 9

°S °° Z

HR

 $\mathbf{Y}$ 

- Vkd3d: D3D 12





Make it easier to use!



lttps://commons.wikimedia.org/wiki/File:HAL9000.sv; 3**C-BY 3.0** 

I'm afraid I can't do that, Dave

## DXVK

ຶ່

O°² Z°

I

 $\mathbf{\Sigma}$ 

#### DX emulator running on Vulkan

- Supports D3D 10 and D3D 11
- Developed by Philip Rebohle with support from Valve Software
- In open source on GitHub

C Sea	arch or jump to		/ Pull re	equests Issues		
📮 doitsujin / <b>dxvk</b>						
<> Code	() Issues 62	រា Pull requests 7	Projects	0 🗉 Wiki		
Vulkan-based D3D11 and D3D10 implementation for Linux / Wine						

#### Extensions created in response to DXVK issues VK\_EXT\_transform\_feedback VK\_EXT\_depth\_clip\_enable VK\_EXT\_host\_query\_reset VK\_EXT\_host\_query\_reset VK\_EXT\_texel\_buffer\_alignment VK\_EXT\_shader\_demote\_to\_helper\_invocation

## **DXVK** in action

#### Valve Software's Proton for Steam Play

- Supports native PC games on Linux
- See https://www.protondb.com
- Powered by DXVK and WINE

With Proton and Steam Play, many Windows games now work on Linux!

SINCE THE RELEASE OF PROTON ONLY 11 MONTHS AGO ...

games reported

5.739

games work

https://www.protondb.com

reports written

https://www.protondb.com



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

51,447 8,833

## Ecosystem, Tools, Developer Support

#### Lots of news on Tools, Validation, SDK

- Validation layer consolidation / configuration
- GPU-assisted validation
- SDK and tools progress



#### **Developer Support**

- Welcome Kris Rose!
- Dedicated DevRel resource for Vulkan / OpenXR



## New functionality highlights

#### VK\_KHR\_shader\_float16\_int8, VK\_KHR\_shader\_float\_controls

- Shader arithmetic on reduced precision types
- Query and override rounding mode, denorm handling, etc

#### VK\_EXT\_descriptor\_indexing

Bindless access to resources

#### VK\_EXT\_buffer\_device\_address (note: KHR version coming soon)

- Reference objects in buffers via 64-bit handles
- E.g., create linked data structures in GPU memory

#### VK\_KHR\_timeline\_semaphore (note: Not quite public; coming soon)

## New functionality since SIGGRAPH 2018

VK\_KHR\_depth\_stencil\_resolve VK\_KHR\_driver\_properties VK\_KHR\_imageless\_framebuffer VK\_KHR\_memory\_model VK\_KHR\_shader\_atomic\_int64 VK\_KHR\_shader\_float16\_int8 VK\_KHR\_shader\_float\_controls VK\_KHR\_surface\_protected\_capabilities VK\_KHR\_uniform\_buffer\_standard\_layout VK\_EXT\_astc\_decode\_mode VK\_EXT\_astc\_texture\_compression\_astc\_hdr VK\_EXT\_buffer\_device\_address VK\_EXT\_calibrated\_timestamps VK\_EXT\_depth\_clip\_enable VK\_EXT\_filter\_cubic VK\_EXT\_fragment\_density\_map

VK\_EXT\_fragment\_shader\_interlock VK\_EXT\_host\_query\_reset VK\_EXT\_index\_type\_uint8 VK\_EXT\_inline\_uniform\_block VK\_EXT\_line\_rasterization VK\_EXT\_memory\_budget VK\_EXT\_memory\_priority VK\_EXT\_pci\_bus\_info VK\_EXT\_pipeline\_creation\_feedback VK\_EXT\_scalar\_block\_layout VK\_EXT\_separate\_stencil\_usage VK\_EXT\_shader\_demote\_to\_helper\_invocation VK\_EXT\_subgroup\_size\_control VK\_EXT\_texel\_buffer\_alignment VK\_EXT\_transform\_feedback VK\_EXT\_validation\_features VK\_EXT\_ycbcr\_image\_arrays

# KHR SOUS GROUP

# Vulkan Video TSG Update

#### Gabor Sines - AMD SIGGRAPH - July 31, 2019

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

© The Khronos<sup>®</sup> Group Inc. 2019 - Page 20

## Vulkan Video TSG Update

Goal

ຶ່

0°2 Z°

2

Т

 $\mathbf{\mathbf{\Sigma}}$ 

- Create a series of extensions for video encode/decode in Vulkan

#### Target release date

- First half 2020

#### Codec roadmap

- H.264, HEVC in first release
- VP9, AV1 will follow in later releases

# KHR SOUS GROUP

# Vulkan ML TSG Update

## Piers Daniell - NVIDIA SIGGRAPH - July 31, 2019

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

(cc) BY

© The Khronos<sup>®</sup> Group Inc. 2019 - Page 22

## Why machine learning in Vulkan?

- Research showcases potential use of machine learning in interactive and high frame rate applications
  - Character animation (phase function neural network, etc.)
  - Full screen image processing (antialiasing, upscaling, inpainting, DLSS, etc.)
  - Non-Player Character bots (AlphaStar, OpenAl Five, etc.)
  - Image generations (GAN, fire & smoke & clouds, etc.)
- Current machine learning solutions have relatively high interop overhead
  - Interop with third party framework (Python TensorFlow, PyTorch, OpenCL, etc.) introduces bubbles where the CPU/GPU are not doing useful work
  - Sharing data with external APIs can be challenging due to difference in memory models and may require additional copies



S O 2 V

2

т



## How to do machine learning in Vulkan?

#### • This is possible already today

- Just use compute shaders to implement the various algebra operations.
- Examples: Tencent/ncnn, Alibaba/MNN, Unity ML, etc.
- Or use compilers which will generate SPIR-V code for you
  - Examples: TVM.AI
- But

2

Т

 $\mathbf{\mathbf{Y}}$ 

- Writing high efficiency layered matrix multiplications, with various activation functions requires some **advanced GPU programming skills**, with **different solutions for different hardware** 



## The Vulkan ML TSG (Task SubGroup)

- A new task subgroup at Khronos has been formed to improve the solution space for machine learning in Vulkan
- Includes representatives from many companies
- Goals

ຶ້

0° 2°

H RO

 $\mathbf{\Sigma}$ 

- Investigate proprietary extensions for inclusion into core Vulkan (VK\_NV\_cooperative\_matrix, etc.)
- Improvements to compute shaders specific to ML needs
- New cross vendor extensions (meta-commands, etc.)
- If you are interested, please reach out to us: <u>pboudier@nvidia.com</u>

BROADCOM<sup>®</sup> Imagination **NVIDIA** Codeplay<sup>®</sup> SAMSUNG arm Google **UALCOMM** 

## VK\_NV\_cooperative\_matrix



- Exposes NVIDIA's Turing Tensor Cores to Vulkan/SPIR-V
- Accelerates large, low-precision matrix multiplies
- Core compute function for deep learning
- FP16 supported today, UINT8/SINT8 coming soon
- Sample code: <u>https://github.com/jeffbolznv/vk\_cooperative\_matrix\_perf</u>
- Performance on NVIDIA TITAN RTX
  - fp16 matrix math with fp16 accumulation: 100 TFLOPS



# KHR SOUS GROUP

# Vulkan Ray Tracing TSG Update

## Yuriy O'Donnell - Epic Games SIGGRAPH - July 31, 2019

## About Ray Tracing Task SubGroup (TSG)

- Part of the Vulkan Working Group
- Collaboration between many hardware and software vendors
- Immediate and future-looking specification design
- Weekly meetings to discuss proposals and track progress
- Face-to-face meetings several times per year



2

Т

## Ray Tracing TSG Goals

• Standard Vulkan API extension for ray tracing on desktop

- Path for ray tracing techniques in existing Vulkan renderers
- Consistent with Vulkan principles
- High-level shading language and SPIR-V extensions
- Validation, debugging and profiling support
- Primary target use cases
  - Real-time rendering

ຶ້

0°2 Z°

2

Т

 $\mathbf{\mathbf{\Sigma}}$ 

- Hybrid ray tracing / rasterization pipelines
- Other possible applications
  - Interactive visualizations and content workflows
  - Offline rendering and baking pipelines

## **Ray Tracing TSG Status**

- Current specification draft based on VK\_NV\_ray\_tracing extension
  - Working towards few additions
    - Capture/replay support
    - Serialization/deserialization of acceleration structures
    - CPU acceleration structure build
    - Indirect build and trace
    - etc.
  - Recursive ray tracing made optional
- Aiming for substantial compatibility with DirectX Ray Tracing (DXR)
  - While keeping to established Vulkan principles
- Release when 2 conformant implementations available

## Give us feedback!

#### **General issues**

https://github.com/KhronosGroup/Vulkan-Ecosystem

#### Spec and functionality issues

https://github.com/KhronosGroup/Vulkan-Docs

#### Validation issues

<u>https://github.com/KhronosGroup/Vulkan-ValidationLayers</u>

## SDK issues

http://vulkan.lunarg.com

## We look forward to hearing from you!