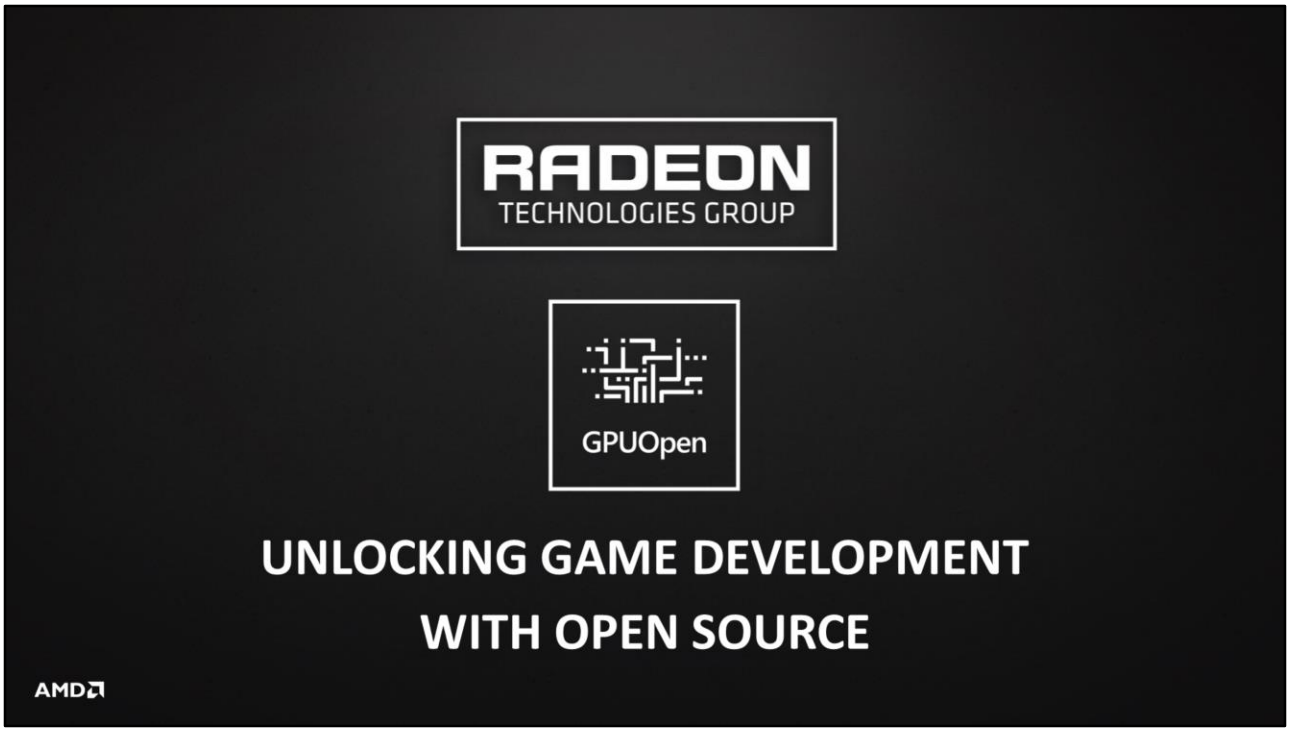
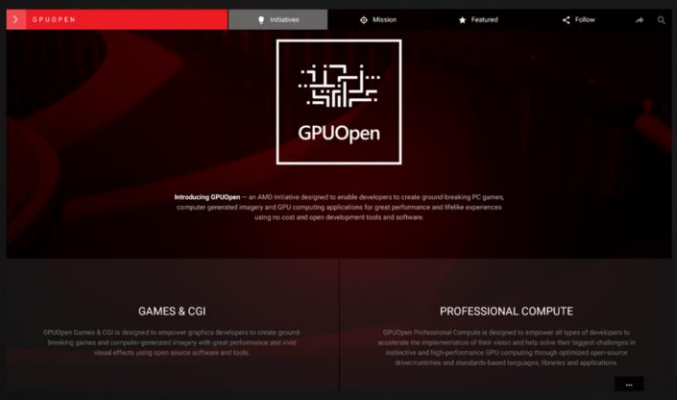

RADEON
TECHNOLOGIES GROUP
GPUOpen
UNLOCKING GAME DEVELOPMENT
WITH OPEN SOURCE
AMD

GPUOPEN – AGENDA



- Introduction and philosophy
- Quick tour
- Currently available
- Blog posts
- Next release
- Special guest

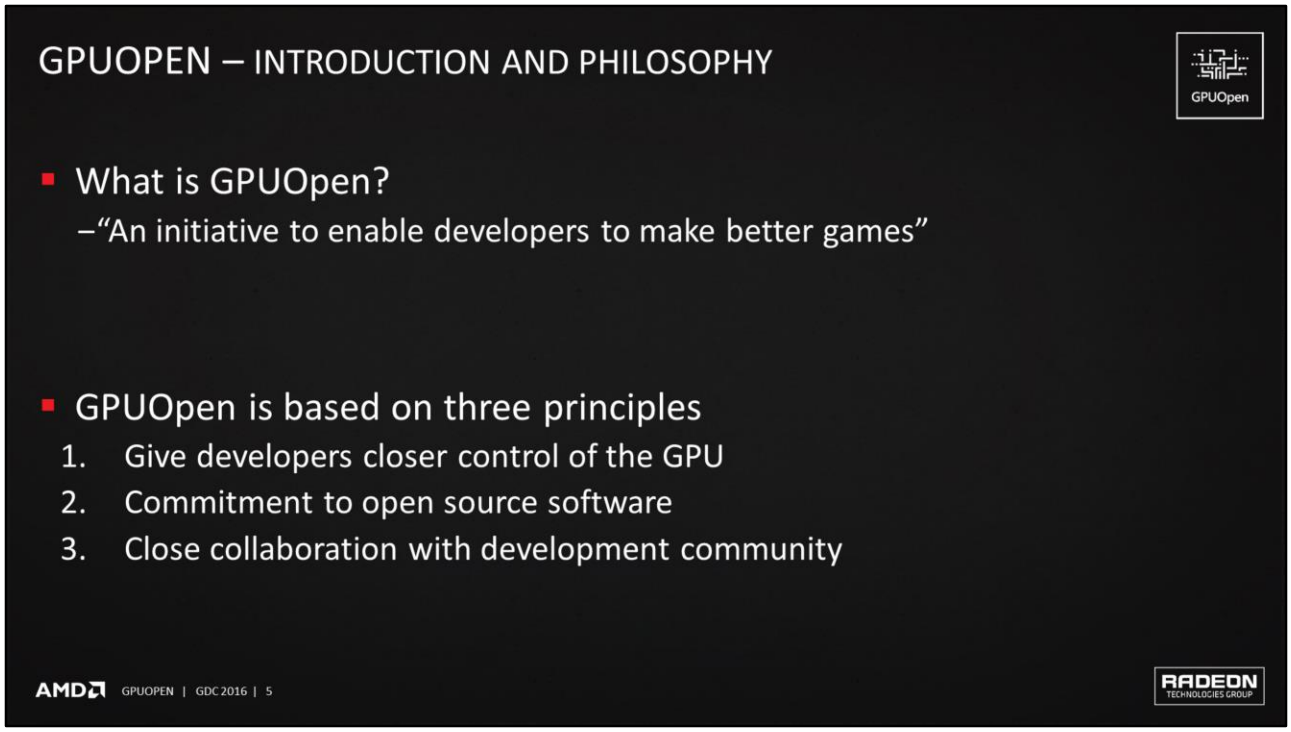
SPECIAL GUEST

- Jean-Normand Bucci
  - –R&D Director for LABS
  - –Eidos Montréal
- GPUOpen philosophy in action

LABS
VIDEOGAMES INNOVATION

eidos
MONTRÉAL


GPUOPEN
INTRODUCTION AND PHILOSOPHY



- What is GPUOpen?
  - Currently, we have divided console & PC development
  - Black box libraries go against the philosophy of game development
  - Game developers are smart and inquisitive
  - Game devs extract the last bit of performance or functionality from a computer system
  - Game devs want to bend the HW to their will
  - This relentless quest is what enabled successive waves of real-time graphics innovation in console and PC games
  - Examples: TressFX porting to XBox One, Dreams by Media Molecule
  - To achieve this developers need knowledge of the HW, get the tools to program this HW efficiently, and need to be able to share this knowledge
  - Game development industry believes strongly in the philosophy of sharing knowledge
    - Industry conferences (GDC, SIGGRAPH, etc.)
    - Carmack's .plan file
  - AMD wants to help developers. Helping developers ultimately benefits end-users through better content.

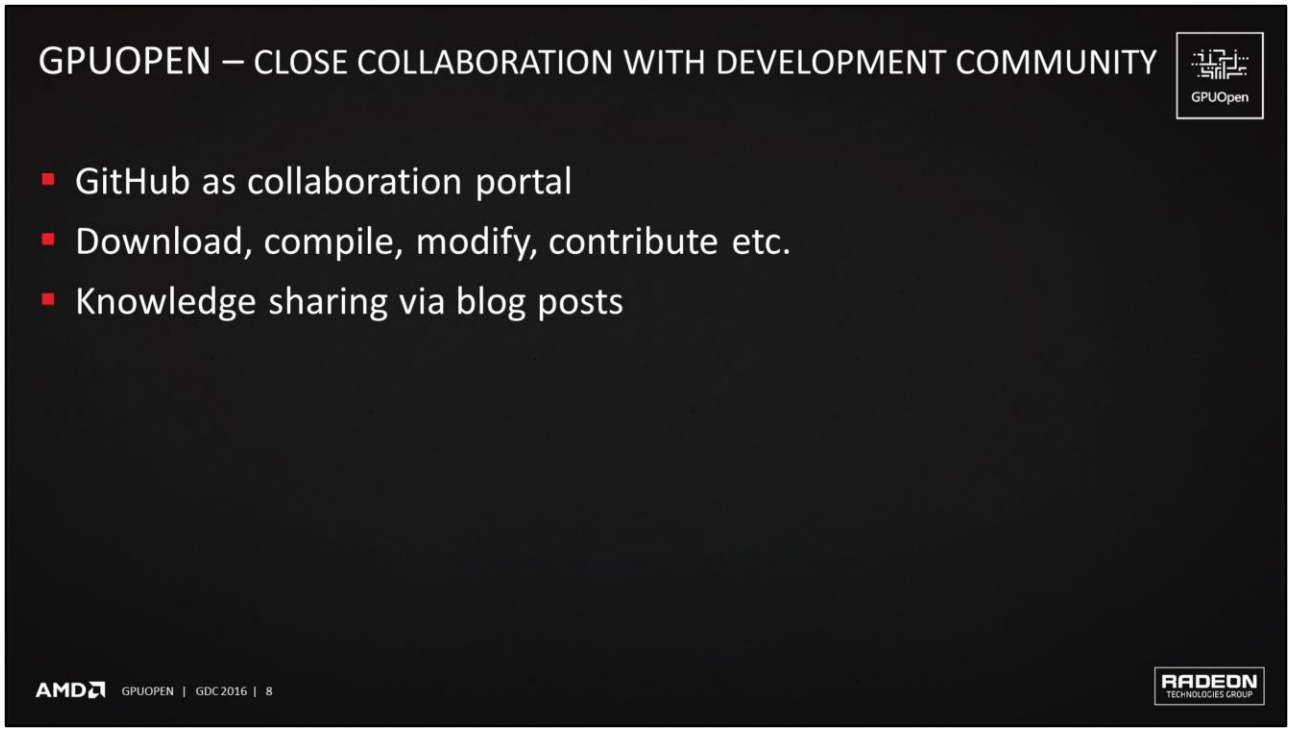
GPUOPEN – BETTER CONTROL OF THE GPU

- Enable GPU features onto the PC platform
- GPU documentation
- Bring console optimizations to PC

GPUOPEN – COMMITMENT TO OPEN SOURCE SOFTWARE

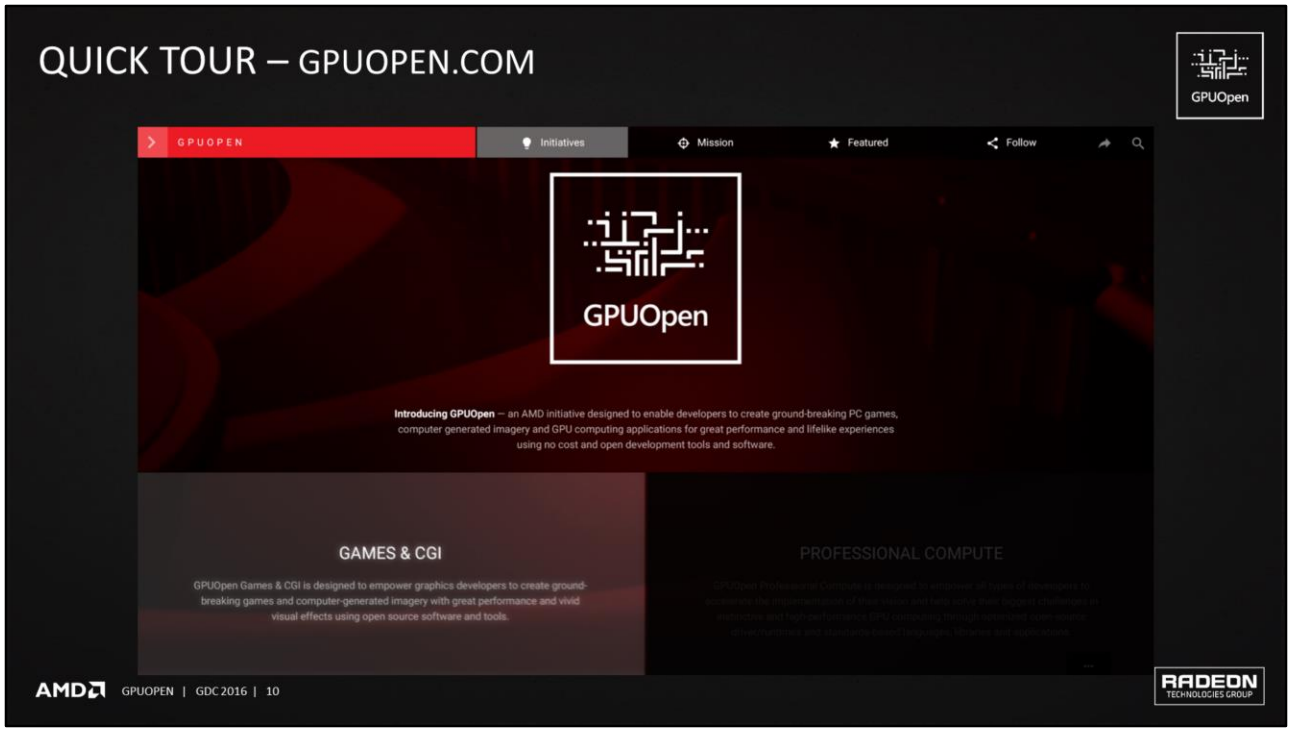
- Gamedevs need to own their code
  - –Stability, expanding, porting, optimizations, learning, etc.
- Shortest way to innovation
- MIT license for most GPUOpen components

- MIT license: No fiddling with licenses – open means open

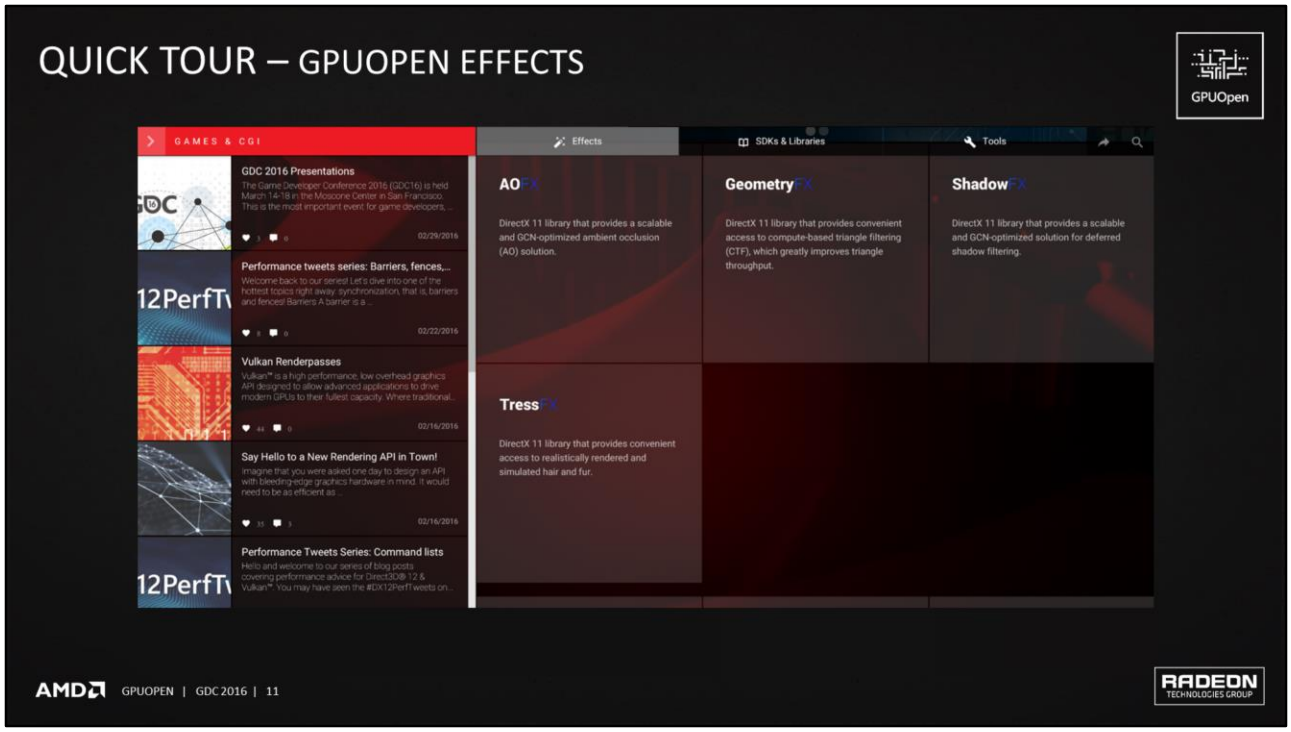


- One blog post a week minimum

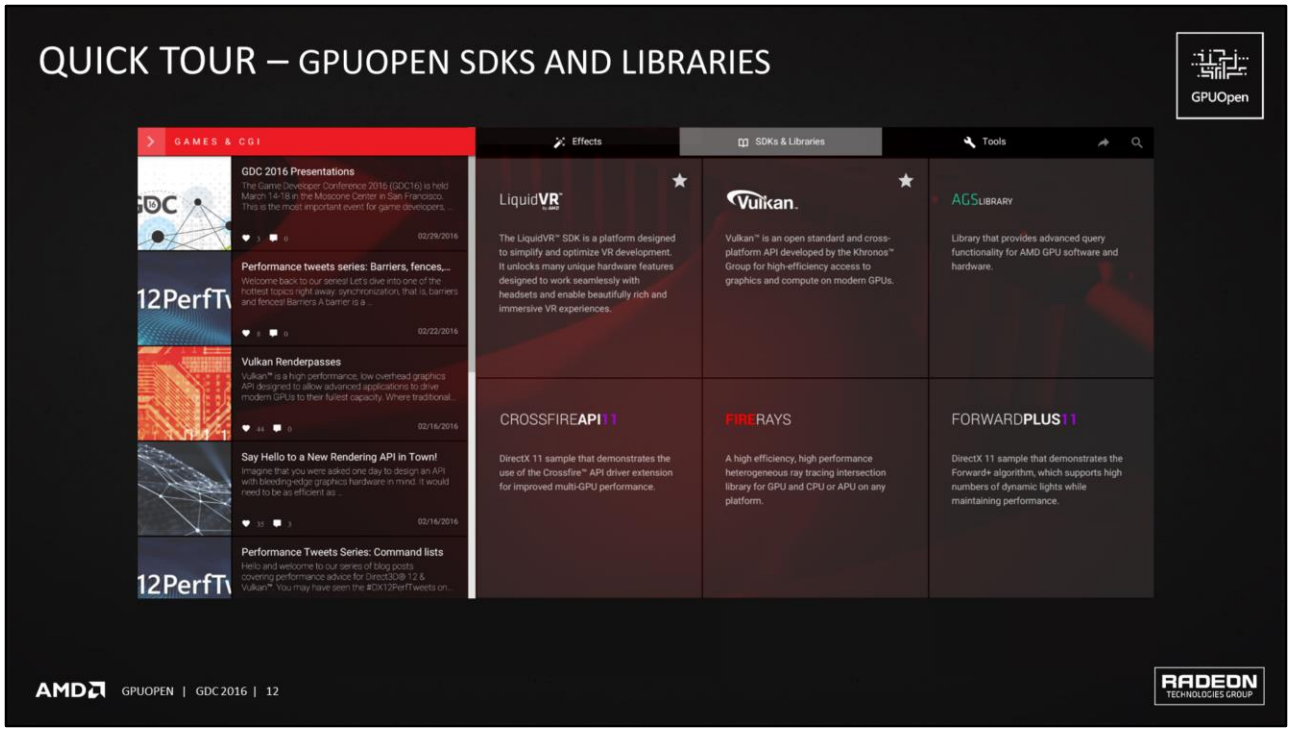

GPUOPEN
QUICK TOUR



- What is GPUOpen?
  - Part of what makes up GPUOpen is a website: gpuopen.com
    - Summary pages for GPUOpen projects
    - Developer-focused blog posts
- I'm going to cover this very quickly
- You all know how to use a browser, and hopefully the navigation on gpuopen.com is intuitive
- But I want to briefly cover how the site is organized
- Two halves. We will be focusing on Games & CGI
- I encourage you to look around the Professional Compute section also
  - But we do not have time to cover that today

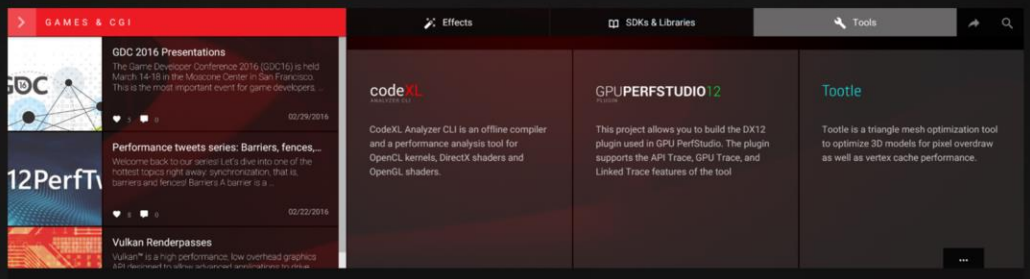


- Three tabs across the top
  - Effects
  - Libraries & SDKs
  - Tools
- This slide shows Effects
  - AOFX, GeometryFX, ShadowFX, TressFX



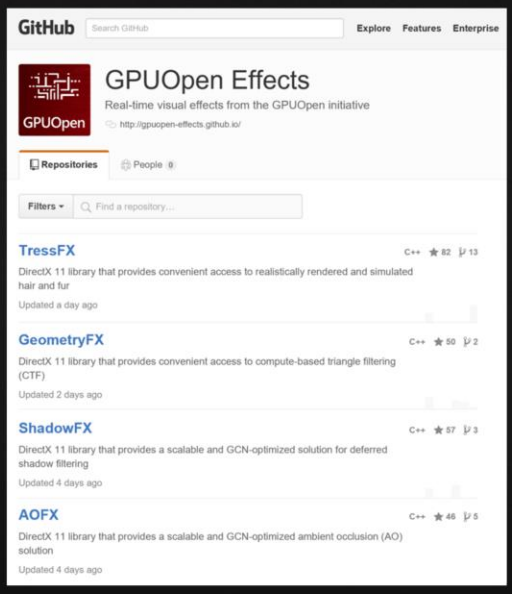
- Libraries & SDKs contains a variety of content

QUICK TOUR – GPUOPEN TOOLS



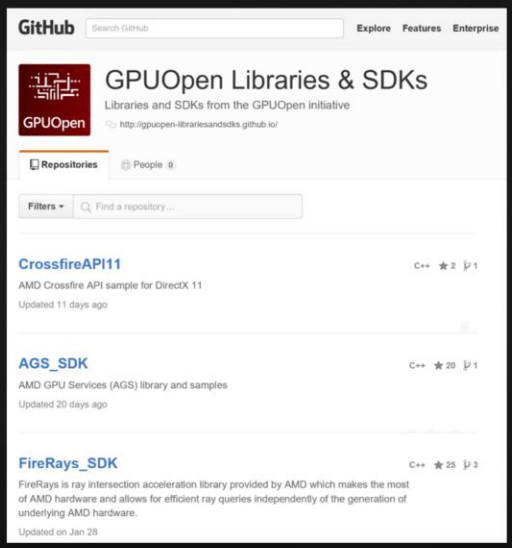
- More will be coming to the Tools section in the next release
- We will cover this later in the talk

QUICK TOUR – GPUOPEN ON GITHUB



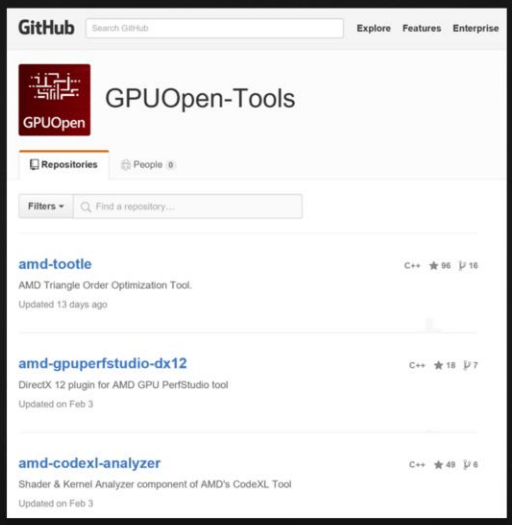
- The three tabs on gpuopen.com each map to an organization on GitHub
  - **Effects**
  - Libraries & SDKs
  - Tools

QUICK TOUR – GPUOPEN ON GITHUB

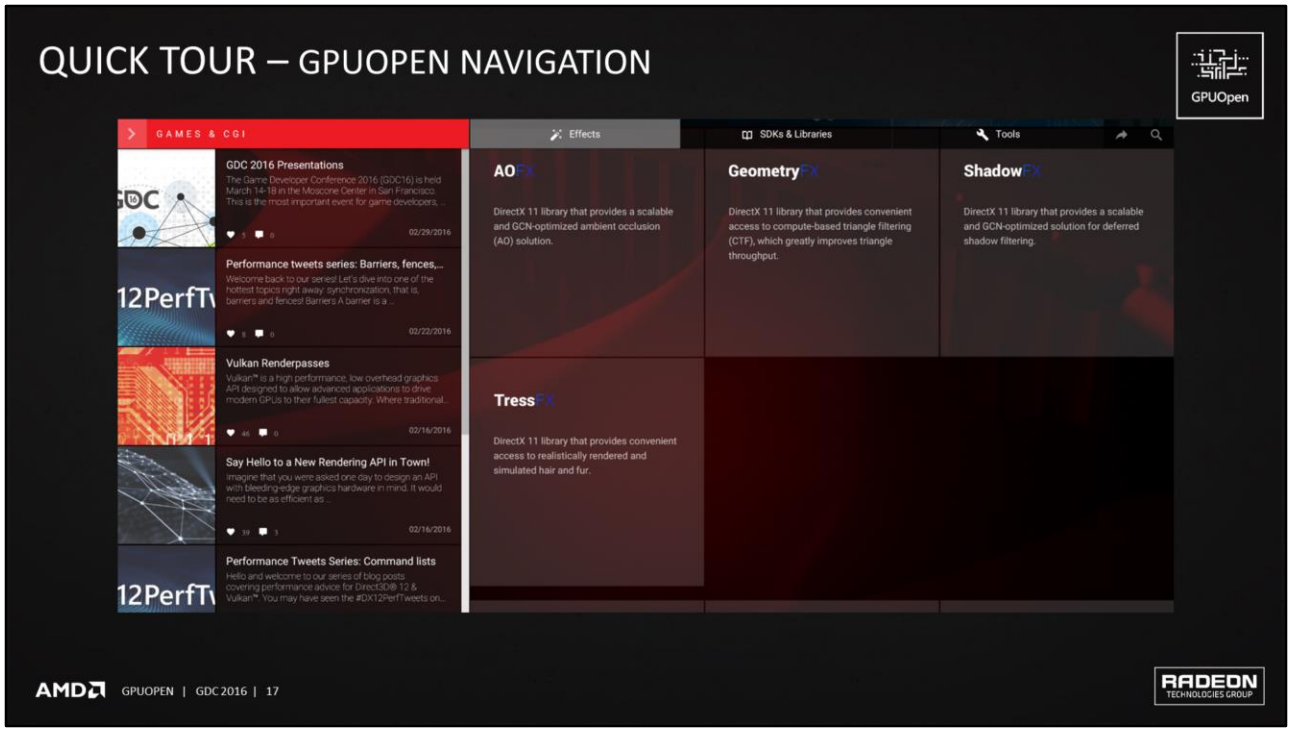


- The three tabs on gpuopen.com each map to an organization on GitHub
  - Effects
  - **Libraries & SDKs**
  - Tools

QUICK TOUR – GPUOPEN ON GITHUB

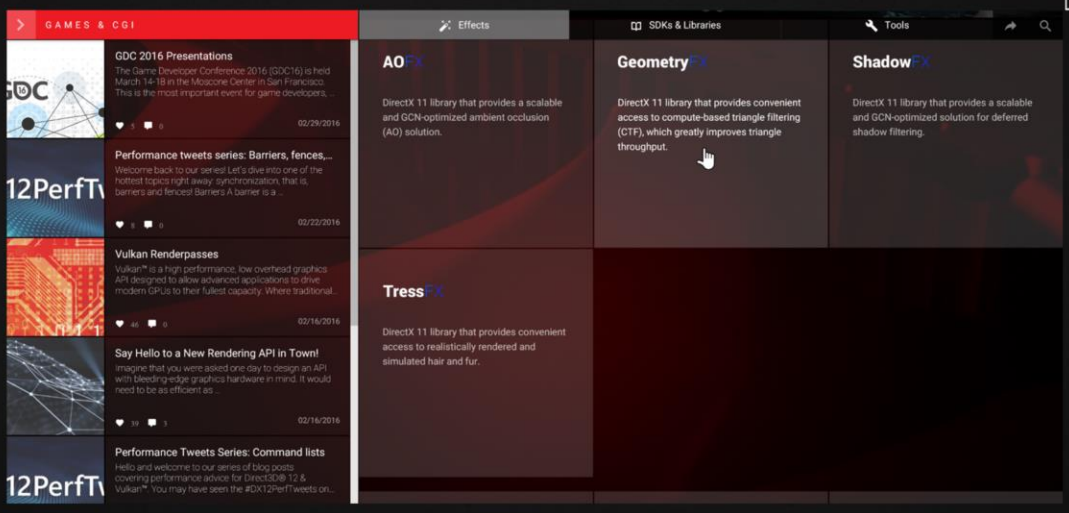


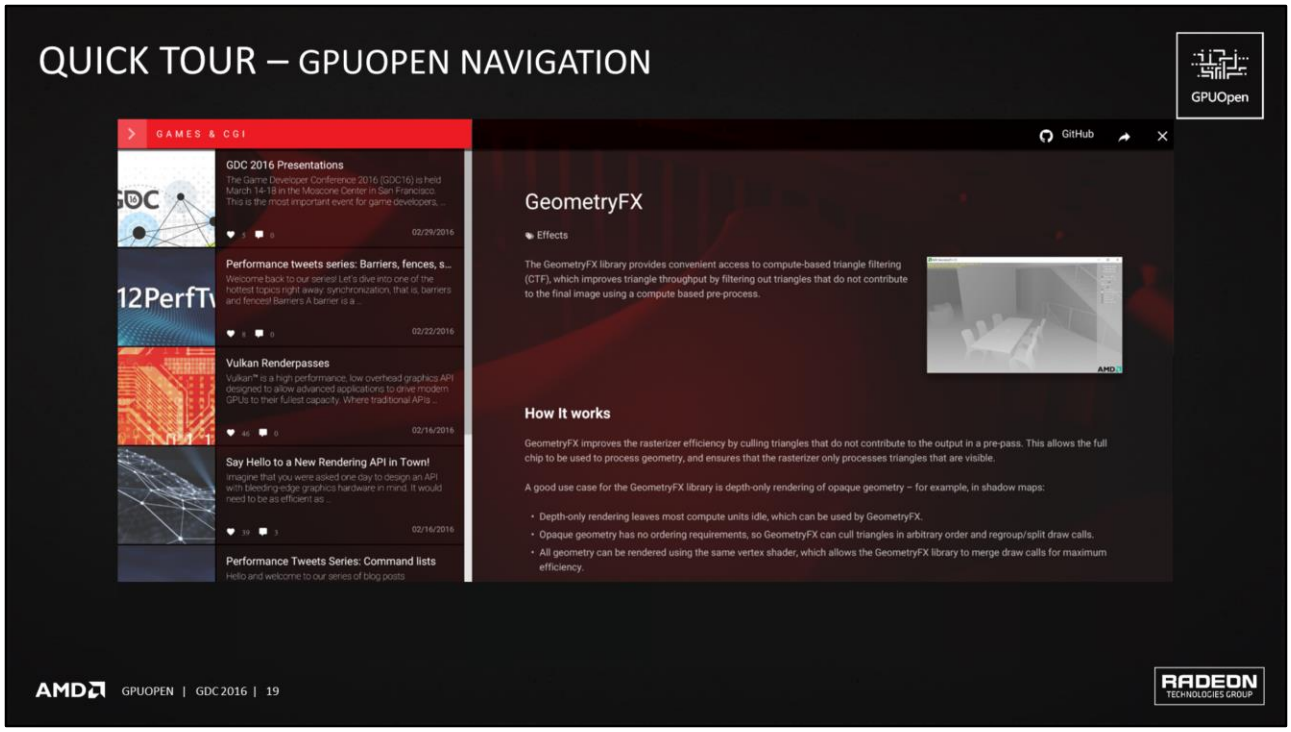
- The three tabs on gpuopen.com each map to an organization on GitHub
  - Effects
  - Libraries & SDKs
  - **Tools**



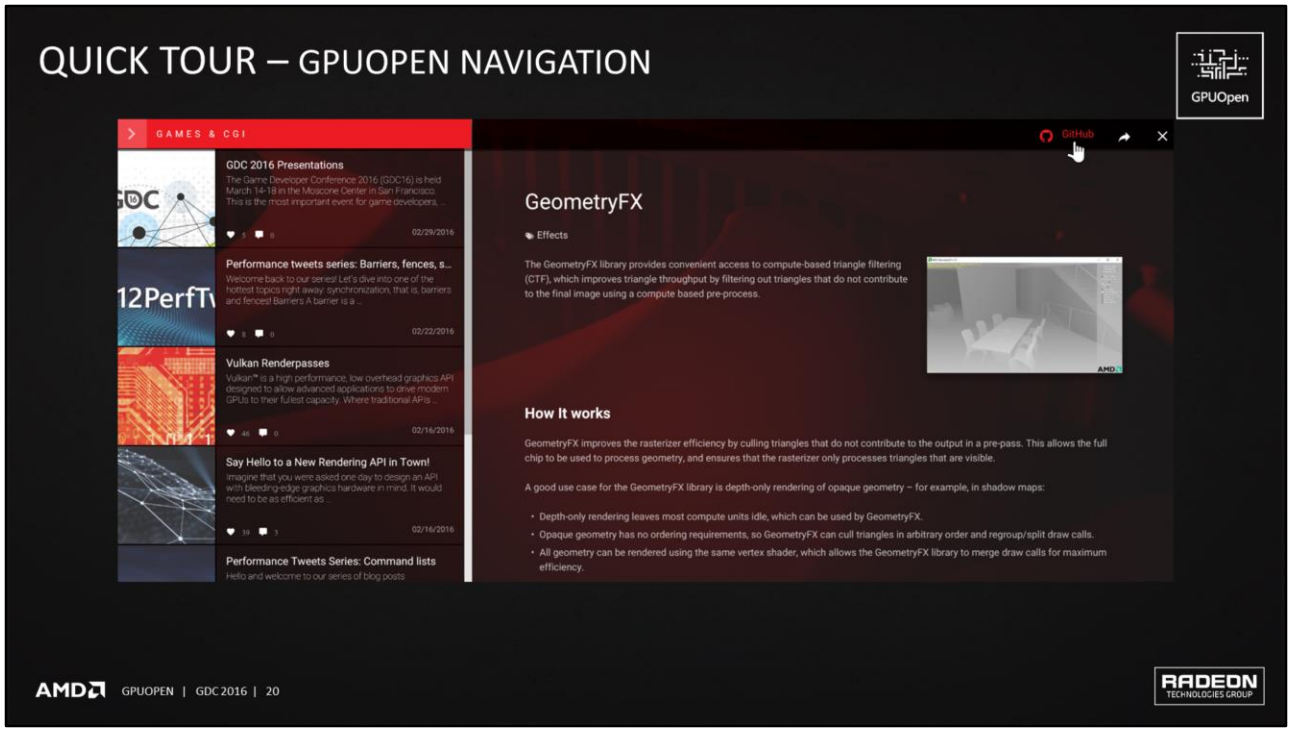
- Okay, back on gpuopen.com, each project has a summary page

QUICK TOUR – GPUOPEN NAVIGATION



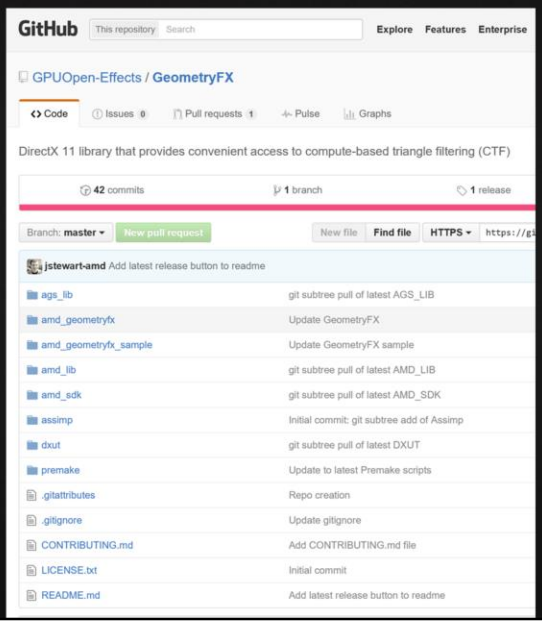


- Here is the summary page for GeometryFX, as an example



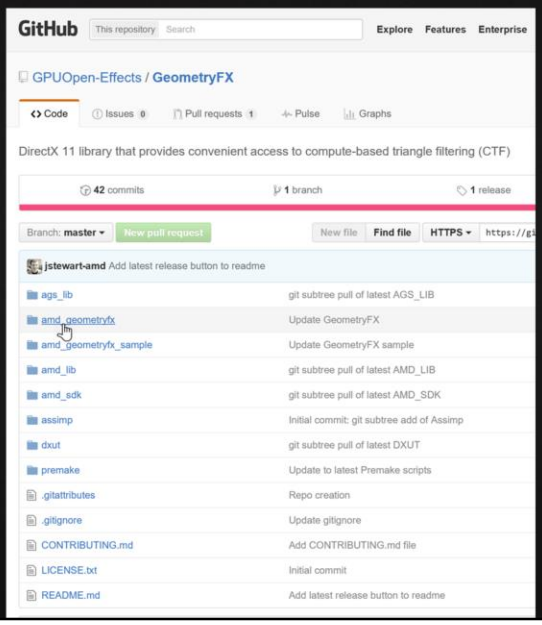
- And then up at the top is a link to the corresponding GitHub repository

QUICK TOUR – GPUOPEN ON GITHUB



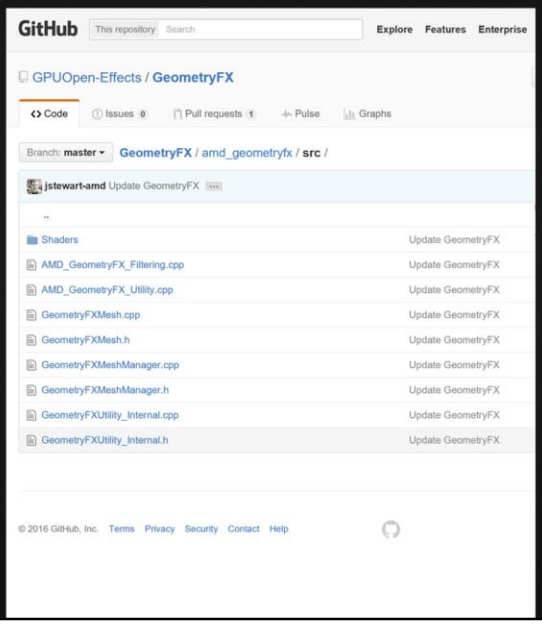
- Full source code
- MIT license
- Pull requests

QUICK TOUR – GPUOPEN ON GITHUB



- Full source code
- MIT license
- Pull requests

QUICK TOUR – GPUOPEN ON GITHUB



- The source is all there, including shaders
- Note, currently all of the Effects have full source, as do many of the repos in Libraries & SDKs. However, some of the content in Libraries & SDKs has source for the samples but still distributes the core library as a DLL
  - We will be open-sourcing more of this going forward
  - Legal review takes time
  - Code cleanup takes time

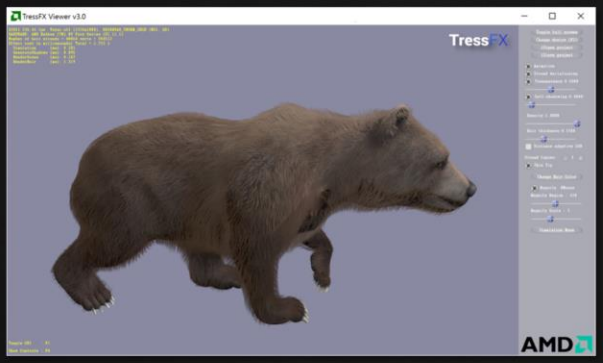
- Let’s quickly run through what is currently available

CURRENTLY AVAILABLE – GEOMETRYFX

- Compute-based per-triangle culling
- Cull geometry at much higher rates than the fixed-function setup
- Up to 30% performance increase in rasterizer-limited situations
  - Depth-only rendering, for example
- Comes with various filters
  - Small primitives, backface, etc.

- Remember the three tabs on gpuopen.com
  - Effects, Libraries and SDKs, Tools
  - These next few slides are Effects

CURRENTLY AVAILABLE – TRESSFX



- Realistic real-time hair and fur
  - Physics simulation on the GPU with compute shaders
  - High-quality rendering
- Full example implementation
- Maya exporter (with source)
- Viewer and runtime library (with source)
- Skinning/animation support
- Fur color from underlying texture

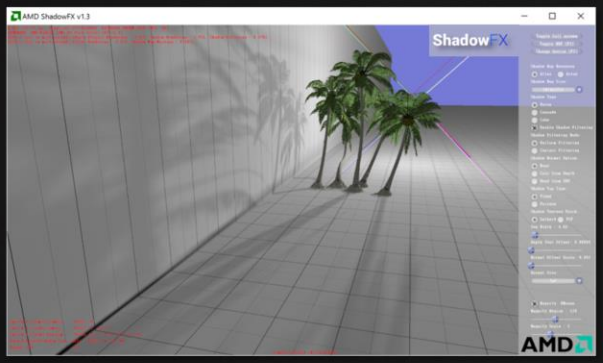
- We've talked about TressFX before at GDC
- We've had an example implementation for a while
- The GPUOpen version is the latest and greatest

CURRENTLY AVAILABLE – AOFX

- Framework for ambient occlusion
- Easily add your own technique
- High Definition Ambient Occlusion (HDAO) kernel
- Pixel shader and compute shader implementations
- Multi-layer approach
- Supports input de-interleaving and downscaling
- Fast, separable, depth-guided bilateral blur

- Very scalable. Exposes many parameters for perf vs. quality tradeoffs
- Multi-layer approach that provides fine details and smooth, alias-free global ambient occlusion
- Takes depth buffer or depth+normal buffers as input
- The AOFX project can also serve as a sandbox for experiments with other AO techniques

CURRENTLY AVAILABLE – SHADOWFX



- Optimized for GCN
- Uniform and contact hardening shadow (CHS) kernels
- Shadow map packing into an atlas or a texture array
- Processing up to 6 shadow maps
  - Union of lights
  - Cascaded light
  - Cube light
- Poisson-distributed and regular sampling patterns

- Optimized for GCN, but runs on everything
  - (The other effects are also optimized for GCN but run on everything)
- The ShadowFX library can also serve as a sandbox for experiments with other shadow filtering techniques

CURRENTLY AVAILABLE – LIQUIDVR™ SDK

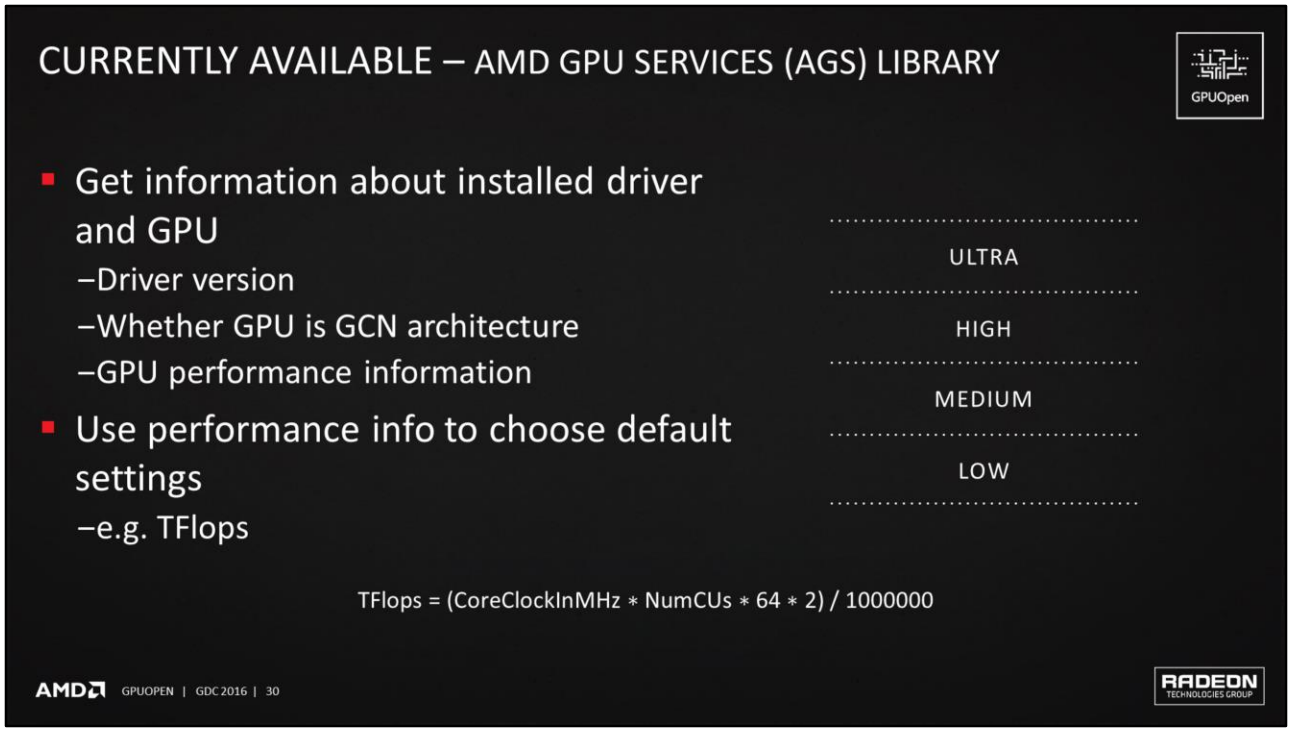
LiquidVR™ by AMD

- Late-Latch
  - Update constant data asynchronously from the CPU to reduce input or sensor latency
- Asynchronous Shaders
  - D3D11 extension similar to async compute in D3D12
- Multi-GPU Affinity
  - Send D3D11 API calls to one or more GPUs via an affinity mask
- Reduce motion-to-photon latency

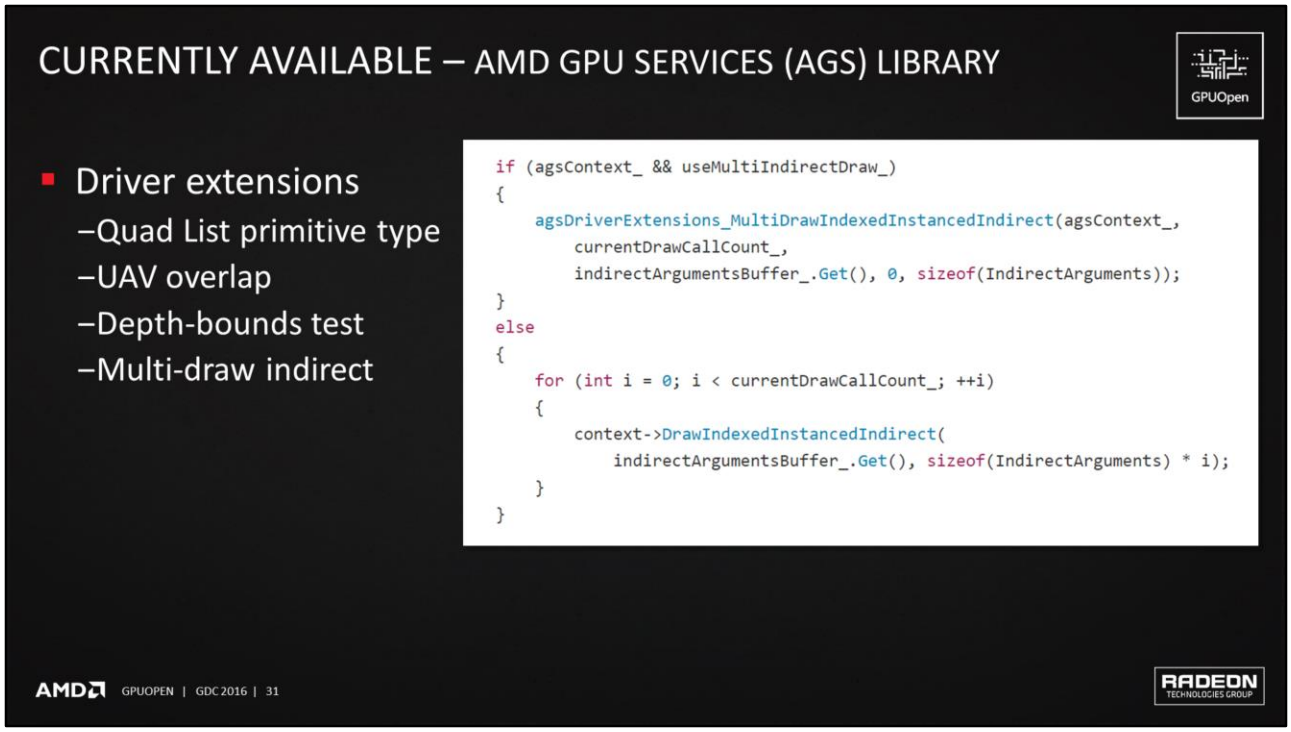
AMD GPUOPEN | GDC 2016 | 29

RADEON TECHNOLOGIES GROUP

- Remember the three tabs on gpuopen.com
  - Effects, Libraries and SDKs, Tools
  - These next few slides are Libraries & SDKs
- Motion-to-photon latency very important in VR
- High def per eye at 90-hz refresh
  - e.g. 2160 x 1200 (1080 x 1200 per eye) at 90
  - Vs. 1920x1080 at 60
  - Half a million (518400) more pixels, or 25% more pixels
  - Only roughly 11 ms to do it vs. 16 ms
  - Actually less than 11 ms in practice (vendor runtime warp, etc.)
- LiquidVR provides a set of features to increase performance and reduce latency
- In addition to what is on the slide, LiquidVR contains the following:
  - GPU-to-GPU Resource Copies
    - Copy resources between GPUs with explicit control over synchronization
  - Direct-to-display
    - HMD vendors are using this to further reduce latency
    - Not exposed in the public LiquidVR API



- AGS is a convenience wrapper to expose AMD-specific functionality in DX11
- Closer access to the GPU, more information about the GPU
- If you've ever wondered how we calculate flops for GCN GPUs, here is the magic formula:
  Clocks per second * NumCUs * 64 instructions per clock per CU * 2 floating point ops per instruction for MAD = floating point ops per second



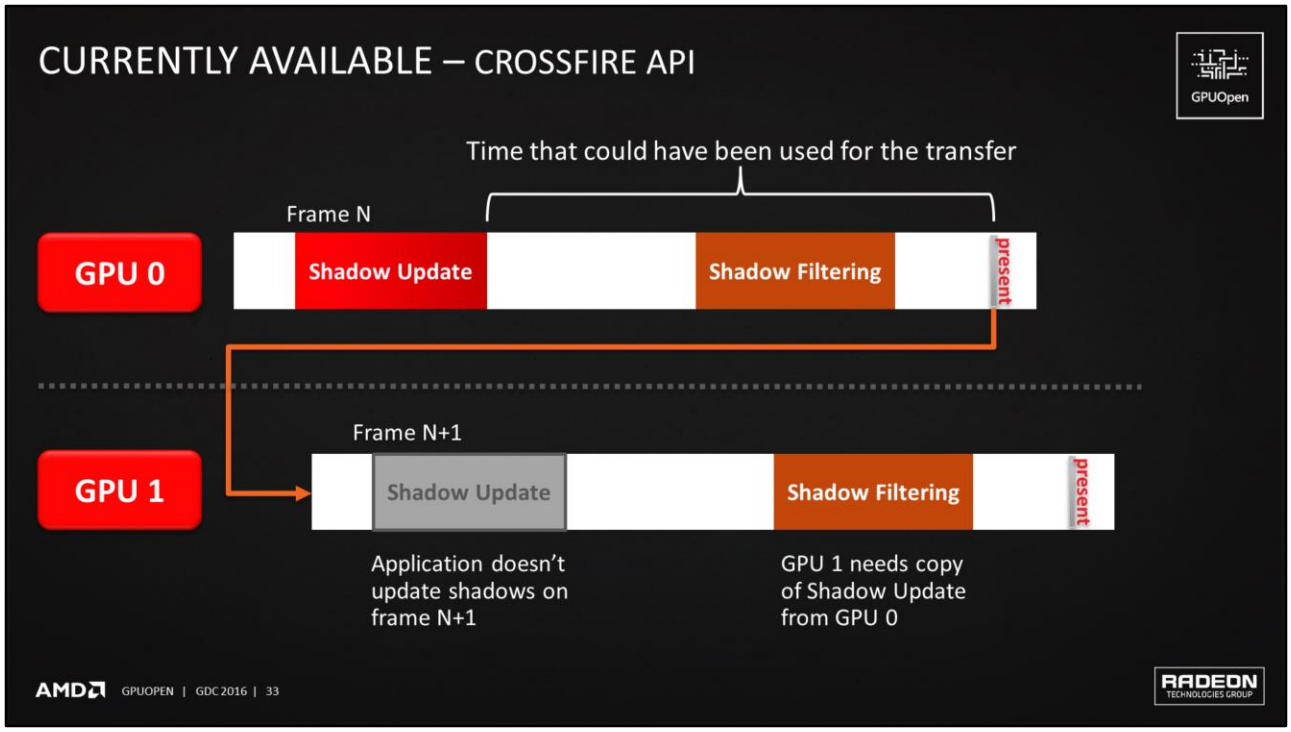
- AGS is how you get at the DX11 driver extensions for AMD
- One of the three principles: closer access to the GPU
  - Or more control of the GPU
- UAV overlap, depth-bounds test, multi-draw indirect
  - Improve performance
- UAV overlap
  - Without this extension, in stock DX11, driver does not have enough information to know if there is a RAW hazard and has to assume the worst
  - Flush, bad for perf
  - This extension allows you to tell the driver it can skip the flush
- Depth-bounds test
  - Optimize deferred lighting
- Multi-draw indirect
  - Reduce draw call count (save DX11 driver CPU overhead)
  - Also the GPU CP can pre-fetch the count so it doesn't stall on every single draw

CURRENTLY AVAILABLE – AMD GPU SERVICES (AGS) LIBRARY

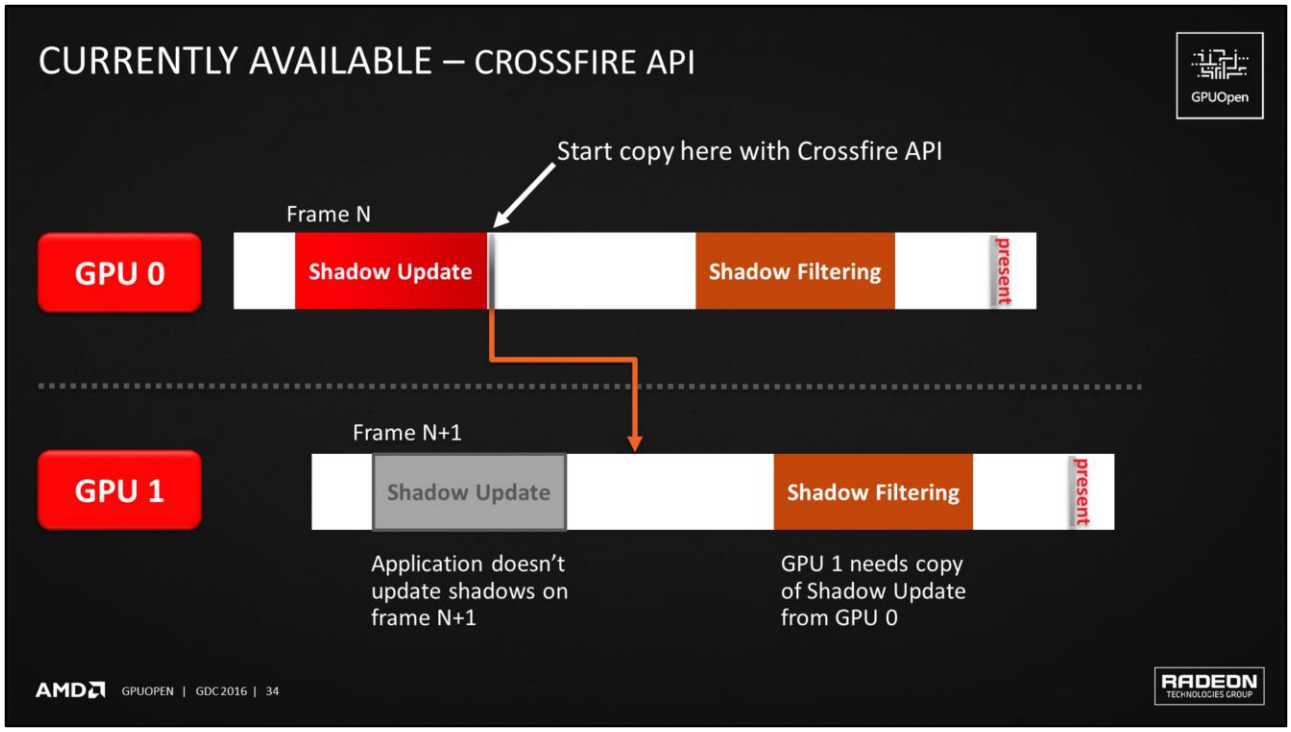
GPUOpen

- User machine configuration info
  - –Crossfire (multi-GPU)
  - –Eyefinity (multi-monitor)
- Explicit Crossfire resource sync API

- AGS is also how you get Crossfire and Eyefinity info
- And we have exposed the Crossfire API as a recent addition



- Speaking of the explicit Crossfire resource sync API, let's go over it because it is going to be important to get good multi-GPU scaling with DX11
- DX11 multi-GPU
  - The driver does alternate-frame rendering (AFR) for you under the hood
  - If the app is fully AFR-friendly, you can get nearly 2x scaling with 2 GPUs
  - But apps are often not fully AFR-friendly (for legitimate reasons)
- Driver can't kick off copy until present



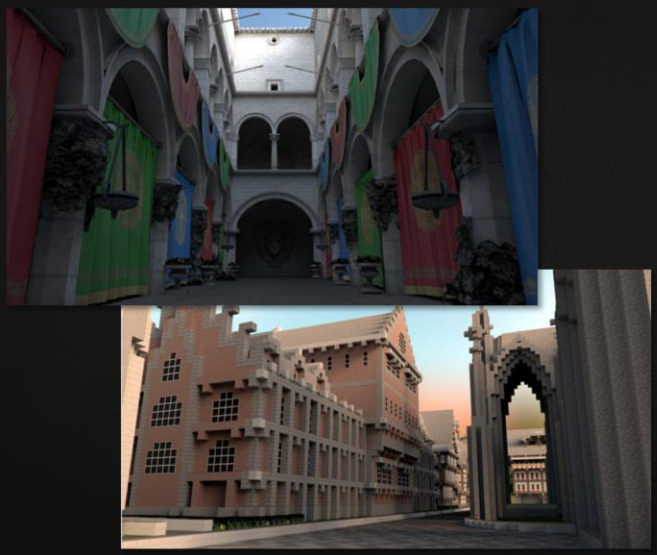
- With the Crossfire API, you can start the copy earlier in the frame, improving multi-GPU performance
- The CrossfireAPI11 sample also contains extensive documentation:
  - *AMD Crossfire guide for Direct3D 11 applications*

CURRENTLY AVAILABLE – SAMPLES

- GPU particles
- Compute-based tiled rendering
  - Forward+
  - Tiled deferred
- Driver extension samples
  - Depth bounds test
- Hello D3D12
- Optimized D3D12 nBody
  - Async Compute
- And many more

- There are lots of individual samples in the Libraries & SDKs section

CURRENTLY AVAILABLE – FIRERAYS 1.X



- Cross-platform, GPU-accelerated ray intersection library
- OpenCL backend
- State of the art traversal
- CPU and GPU BVH rebuild
- Motion blur support
- Instancing support
- Open-source example renderer
- Multi GPU support

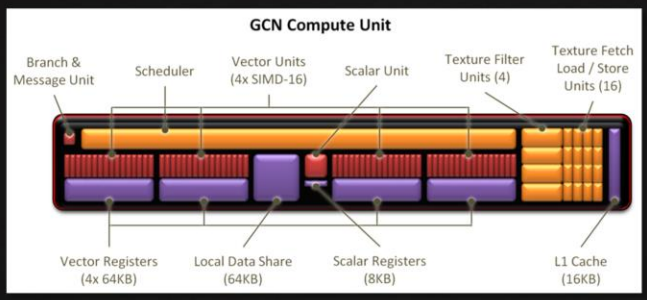
- GPU accelerate your offline baking, e.g. GI
- BVH = Bounding Volume Hierarchy
- Will have more from the tools team later in this session about what is coming in the next release

CURRENTLY AVAILABLE – FIRERAYS 1.X

- Cross-platform, GPU-accelerated ray intersection library
- OpenCL backend
- State of the art traversal
- CPU and GPU BVH rebuild
- Motion blur support
- Instancing support
- Open-source example renderer
- **Multi GPU support**

- Regarding multi-GPU support, the example render is from the 3dsmax plug-in and shows a 1.75x perf increase going from 1 to 2 GPUs
- And the GPU-accelerated render, even with a single GPU, will be much faster than rendering on the GPU
- So again, take a look at these tools to accelerate any offline rendering you have in your pipelines

CURRENTLY AVAILABLE – TOOLS



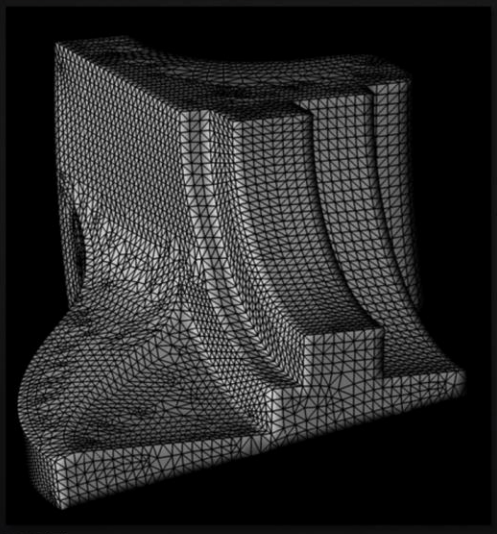
Device	SGPRsUsed	MaxSGPRs	VGPRsUsed	MaxVGPRs	UsedLDS	MaxLDS	usedScratchBytes
Hawaii	28	102	7	256	16	32768	0

GCN VGPR Count	<=24	28	32	36	40	48	64	84	<= 128	> 128
Max Waves/SIMD	10 ☺	9	8	7	6	5	4	3	2 ☹	1 ☹

- CodeXL Analyzer CLI
  - –Command-line tool to analyze GCN shaders
  - –Hardware resource usage
  - –Can produce AMD ISA source listing
- Create your own GPU PerfStudio DirectX® 12 plugin

- We've covered two of the three tabs
  - Effects, Libraries & SDKs
- Now the third tab: Tools
- Pictured on the left is the venerable GCN compute unit
- CodeXL Analyzer CLI lets you get resource usage stats and ISA source listings for your shaders

CURRENTLY AVAILABLE – TOOLS

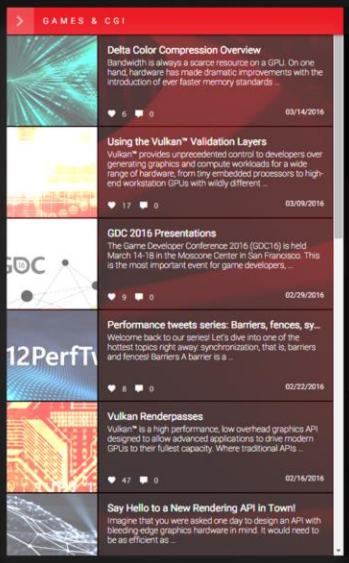


- Tootle
- Reorder triangles (both vertex and index buffer) to optimize for:
  - –Post-transform vertex cache
  - –Early Z-culling hardware
  - –Hardware cache line size
- Applicable to any modern GPU
- Easy integration into mesh pre-processing tool chain

- In case you are wondering, the name Tootle is inspired by the pronunciation of T. O. O. Tool (Triangle Order Optimization Tool)
- Tootle integrated into the asset pipeline for Warframe by Digital Extremes
- We are starting to see it integrated into the asset pipelines at other studios

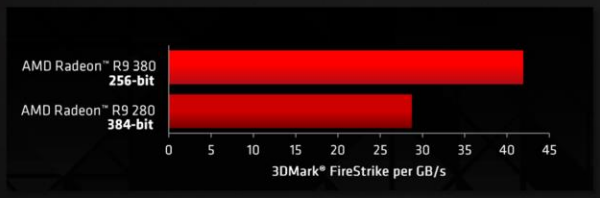
- One of the three GPUOpen princples was to engage with the developer community
- Technical blog posts are one of the ways we are doing this

BLOG POSTS



- Back on gpuopen.com, over to the left, there is a list of blog posts
- HW information, programming tricks, performance advice, etc.

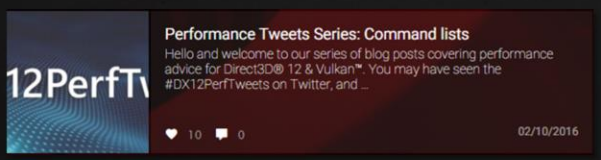
BLOG POSTS – LEADING INDUSTRY EXPERTS

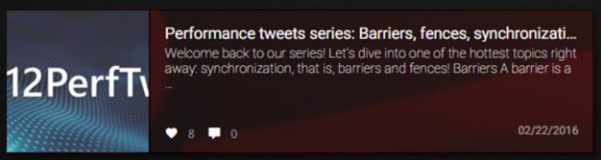


- GPU hardware blogs written by the GPU hardware architects
- Hardware details
- Dos and Don'ts
- Example: Organize G-Buffer data to maximize compression
  - –Highly correlated bits in the Most Significant Bits (MSBs)
  - –Noisy data in the Least Significant Bits (LSBs)

- Chris Brennan is a GPU Hardware Architect and drives the fixed function graphics pipelines and compression
- Give PC developers more knowledge of the GPU
- Color render targets are stored in a lossless compressed format
- The shader core can read the compressed color
- If arbitrarily bit-packing fields into a G-Buffer, put highly correlated bits in the Most Significant Bits (MSBs) and noisy data in the Least Significant Bits (LSBs) of each channel. This will compress better because it responds similarly to typical data patterns.

BLOG POSTS – DX12 PERFORMANCE ADVICE





- Command lists, barriers, fences, etc.
- Summarize and expand upon the DX12 perf tweets
- Example: Don't do read-to-read barriers
  - –Get the resource in the right state the first time

BLOG POSTS – VULKAN

GPUOpen

Vulkan™

Vulkan Renderpasses

Vulkan™ is a high performance, low overhead graphics API designed to allow advanced applications to drive modern GPUs to their fullest capacity. Where traditional APIs ...

♥ 47 💬 0 02/16/2016

Using the Vulkan™ Validation Layers

Vulkan™ provides unprecedented control to developers over generating graphics and compute workloads for a wide range of hardware, from tiny embedded processors to high-end workstation GPUs with wildly different ...

♥ 17 💬 0 03/09/2016

- Vulkan overview
- Details on renderpasses, validation layers, etc.
- Code snippets

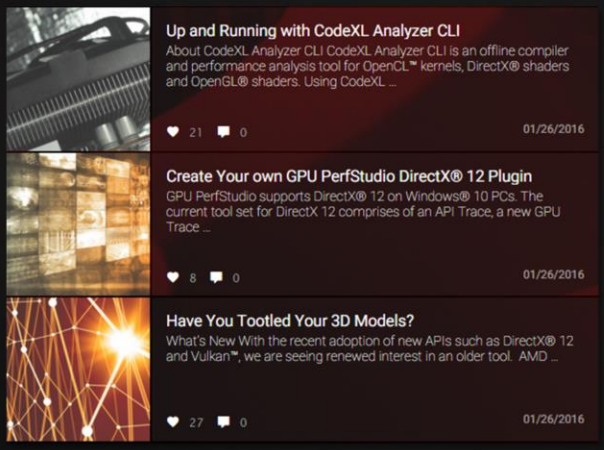
```
#ifdef MY_DEBUG_BUILD_MACRO
    /* Load VK_EXT_debug_report entry points in debug builds */
    PFN_vkCreateDebugReportCallbackEXT vkCreateDebugReportCallbackEXT =
        reinterpret_cast<PFN_vkCreateDebugReportCallbackEXT>
            (vkGetInstanceProcAddr(instance, "vkCreateDebugReportCallbackEXT"));
    PFN_vkDebugReportMessageEXT vkDebugReportMessageEXT =
        reinterpret_cast<PFN_vkDebugReportMessageEXT>
            (vkGetInstanceProcAddr(instance, "vkDebugReportMessageEXT"));
    PFN_vkDestroyDebugReportCallbackEXT vkDestroyDebugReportCallbackEXT =
        reinterpret_cast<PFN_vkDestroyDebugReportCallbackEXT>
            (vkGetInstanceProcAddr(instance, "vkDestroyDebugReportCallbackEXT"));
#endif
```

AMD GPUOPEN | GDC 2016 | 44

RADEON TECHNOLOGIES GROUP

- Written by those directly involved in Vulkan
  - Directly involved in defining Vulkan
  - Directly involved in implementing the AMD driver for Vulkan

BLOG POSTS – TOOLS

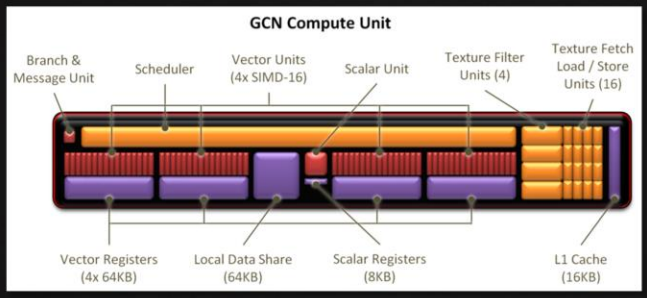


- CodeXL Analyzer CLI
  - –Analyze your GCN shaders
- GPU PerfStudio DirectX® 12 plugin
- Tootle
  - –Optimize your meshes

- There are blog posts corresponding to each of the three tools projects covered in the “Currently Available” section

- This “next release” is coming in the mid-April 2016 timeframe

NEXT RELEASE – SHADER EXTENSIONS

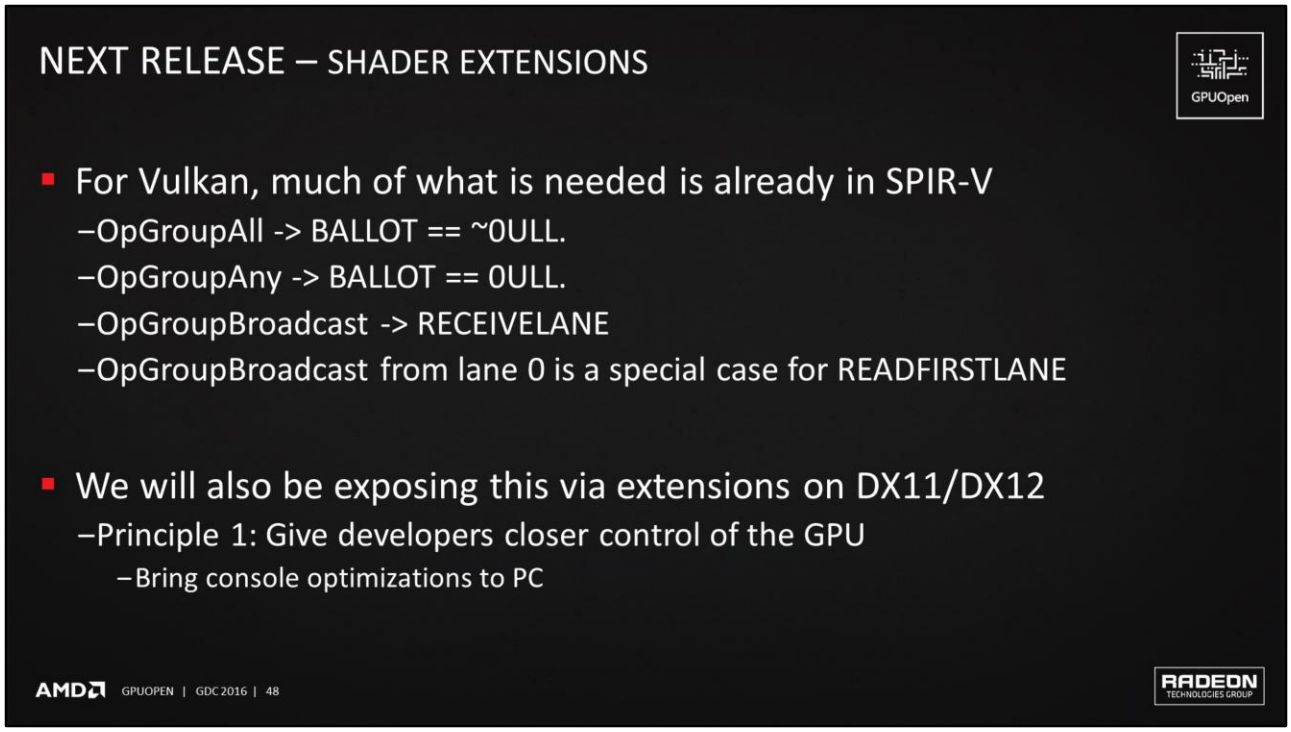


Device	SGPRsUsed	MaxSGPRs	VGPRsUsed	MaxVGPRs	UsedLDS	MaxLDS	usedScratchBytes
Hawaii	28	102	7	256	16	32768	0

GCN VGPR Count	<=24	28	32	36	40	48	64	84	<= 128	> 128
Max Waves/SIMD	10 ☺	9	8	7	6	5	4	3	2 ☹	1 ☹

- readfirstlane
  - Can significantly reduce VGPR usage
  - Tells the compiler a value is uniform across a wavefront
  - Allows it to go into an SGPR, reducing VGPR pressure
- ballot
  - Can improve performance a lot in highly uniform control flow cases
  - Provides access to the execution mask across a wavefront
  - if (ballot (x)) { foo; } else { bar; }

- Once again, the venerable GCN compute unit
- Branching in a CU, e.g. if-else
  - 64-bit execution mask
  - If all threads in a wavefront take the same path, e.g. the if case, the CU can skip the work of the else case
  - But this happens at execution time. The compiler has to assume some threads might take the if and some the else
  - And so the compiler has to allocate resources for both the if and else
- Ballot returns a bitfield for the execution mask
- If your algorithm is structured so that you can say, “only execute this branch of the if-else when all threads evaluate the conditional to the same value”, then you can use ballot
- In example code, if all threads evaluate x to true, do foo, else do bar
  - Now the compiler knows foo and bar are *exclusive*
  - So if foo needs 4 registers and bar needs 4, it can allocate 4 instead of 8 for if (x) { foo; } else { bar; }



- These useful shader optimizations are available on consoles. How can you get to them on PC?
- Much of what is needed is already in Vulkan SPIR-V
  - The details of what specific GLSL to write to get at this are still being finalized
- For DX11/DX12, we will be exposing this via extensions

NEXT RELEASE – SHADER EXTENSIONS

- v_readfirstlane_b32
- v_readlane_b32
- LaneId
- ds_swizzle_b32
- Ballot
- v_mbcnt_lo_u32_b32/v_mbcnt_hi_u32_b32
- v_bcnt_u32_b32
- v_{min,med,max}_3[UF]
- Direct access to i/j barycentrics and vertex parameters in pixel shader

RADEON TECHNOLOGIES GROUP

- Here is the full list of what will be exposed

NEXT RELEASE – GEOMETRYFX

- Improved culling
- Cluster culling
  - Removing whole chunks of geometry
- Integration of shader extensions
  - Ballot
  - if (ballot (cull) != 0 && (cullFlags & CULL_SMALL_PRITIMIVES))

- Remember the three tabs
  - Effects, Libraries and SDKs, Tools
  - GeometryFX is Effects
- Also will be continuing work on TressFX
  - Optimizations
- For Libraries & SDKs, we will have Vulkan samples and more D3D12 samples
- And we will be continuing to open source more code

WORKSTATION – FIRERAYS 2.0

GPUOpen

- New FireRays API
- Fully open-source!
- New features
  - Improved motion-blur
  - Subdivision
  - Out-of-core
- New demos (SSS, PM)
- Integration into FireRender Framework
- Vulkan backend

WORKSTATION – FIRERENDER

- Multi back end framework for the API
  - Optimized path tracer backend
  - Rasterizer OpenGL
  - Rasterizer Vulcan
  - Open-source FireRays backend
- Vulkan framework engine
- Cluster rendering support

COMING SOON – MORE DEVELOPER TOOLS

- Developer tools software that will be open-sourced and hosted on GitHub as part of GPUOpen:
  - CodeXL
  - GPA
  - Compress
- Everyone is welcome to join the development and benefit from the tools
  - GitHub pull requests
- We will continue developing new features and fixing bugs on the public GitHub repositories
- Our tools are continuously tested and verified on:
  - Windows 7 64-bit, 8.1 64-bit and 10 64-bit
  - Ubuntu 64-bit
  - Red-Hat 64-bit

Tools & technologies to enhance the Open Source ecosystem

COMING SOON – CODEXL

- CodeXL is a tool suite which helps SW developers get the best performance on AMD CPUs and GPUs
- Debug, Profile and Analyze applications on local and remote hosts
  - Combined Debugger of Host and GPU code
  - DX12 Frame Analysis
  - Shader Analyzer
  - CPU Profiler
  - Power Profiler
- Multiple platforms and Operating Systems
  - Standalone application for Windows® and Linux®
  - Integrated into Microsoft® Visual Studio®
  - Linux and Windows feature parity
- Multiple new releases every year

CODE XL

RADEON TECHNOLOGIES GROUP

COMING SOON – GPU PERFORMANCE API (GPA)

GPUOpen

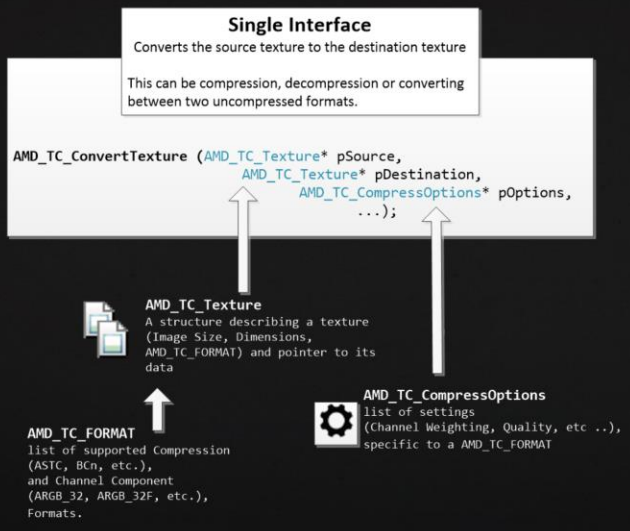
- Library for accessing GPU performance counters on AMD Radeon graphics cards and APUs
- Analyze the performance and execution characteristics of applications that use the GPU
- Provides derived counters based on raw HW performance counters
- Supports Windows and Linux
- Used by GPU PerfStudio and CodeXL
- Easy to incorporate into third party applications
- Supports DirectX11, OpenGL, OpenGL ES, and OpenCL applications

COMING SOON – COMPRESS

- What will be provided
  - GUI Tool source
  - Command Line Tool Source
  - Compress SDK Source Code
  - Common Code used for GUI and Command Line
- MIT Licensed
- Available on GPUOpen Tools on GitHub
  - https://github.com/GPUOpen-Tools/

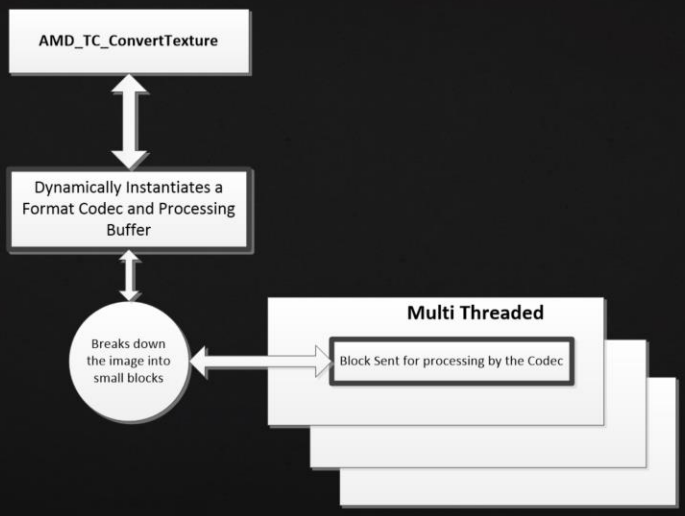
COMPRESS

SIMPLIFIED API FOR APPLICATION INTERGRATION



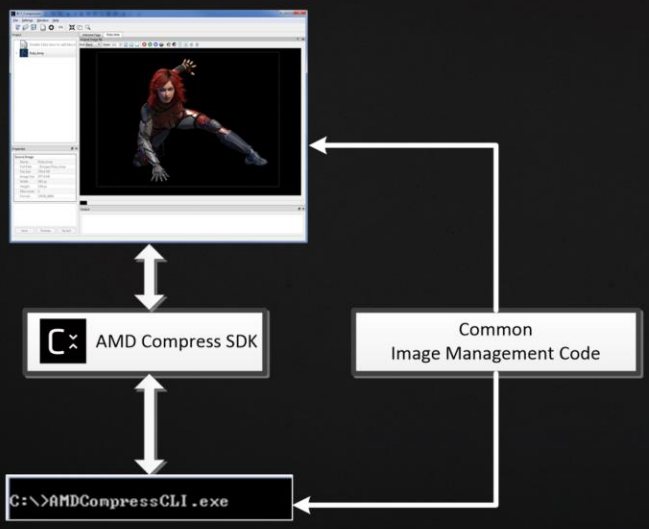
COMPRESS

GET AN INSIGHT AS TO THE INNER WORKINGS OF THE SDK



COMPRESS

COMMON CODE FOR TOOLS DEVELOPMENT



Both GUI and Command Line tools share a common set of management code that manages Loading, Saving Image Files, Perform Analysis, Generate MIP levels etc.

The SDK is used to process images using the single API interface

Developers can create any number of new tools with specialized functionality using the open source SDK


GPUOPEN
SPECIAL GUEST

SPECIAL GUEST

- Jean-Normand Bucci
  - –R&D Director for LABS
  - –Eidos Montréal
- GPUOpen philosophy in action

LABS
VIDEOGAMES INNOVATION

eidos
MONTRÉAL

- In the live presentation, Jean-Normand Bucci came on to talk about how they took open-source TressFX and improved it and incorporated it into their PureHair technology

ENHANCE!

- JN's content is not included in these slides


GPUOPEN
SUMMARY

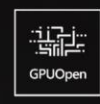
GPUOPEN – SUMMARY

- GPUOpen is based on three principles
  1. Give PC developers more control of the GPU
  2. Commit to open source software
  3. Engage and collaborate with the developer community
- Currently available
  - –Effects: GeometryFX, TressFX, AOFX, ShadowFX
  - –Libraries & SDKs: LiquidVR, AGS, Crossfire API, FireRays, samples
  - –Tools: CodeXL Analyzer CLI, Tootle, etc.

GPUOPEN – SUMMARY

- Blogs
  - –Lots of good info from industry experts
  - –New blogs added regularly
- Next release
  - –Shader extensions
  - –Updated effects
  - –FireRays 2.0 and FireRender
  - –Full CodeXL open source, GPU Performance API (GPA), Compress
- GPUOpen principles in action
  - –From TressFX to PureHair

DISCLAIMER & ATTRIBUTION



The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions and typographical errors.

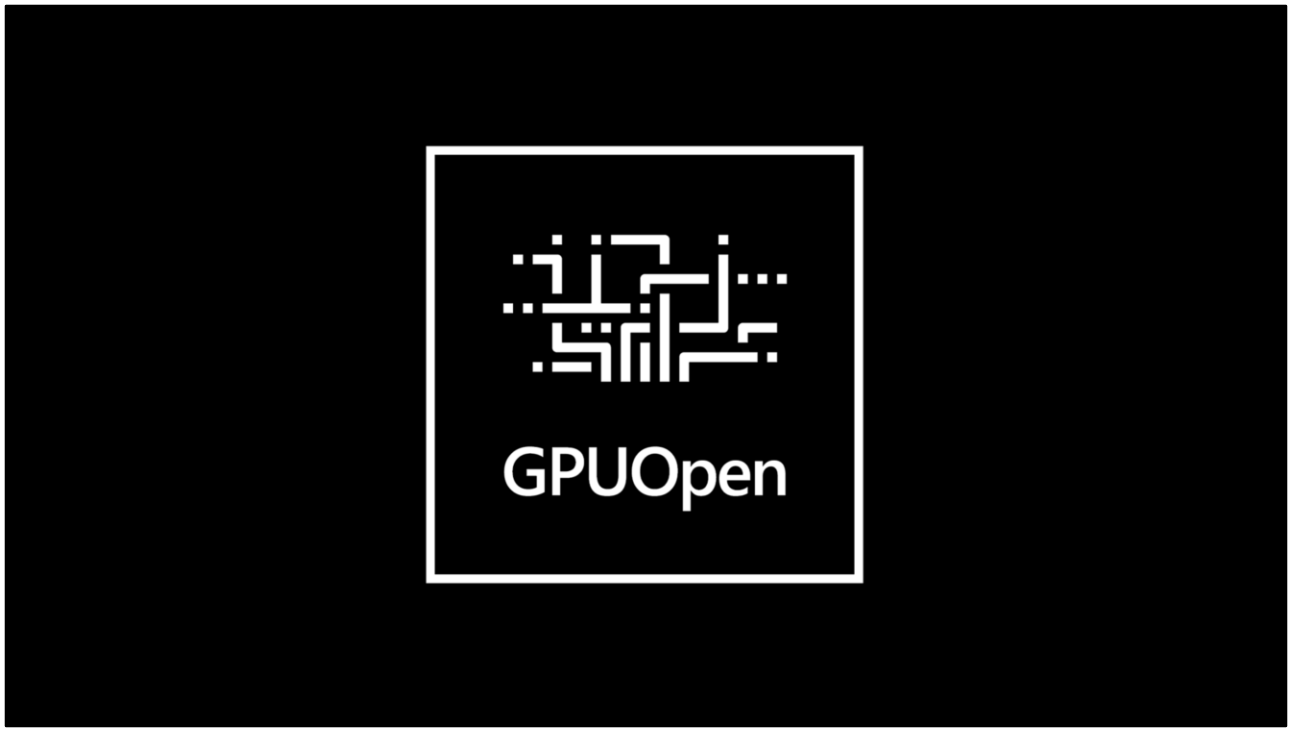
The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION.

AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY DIRECT, INDIRECT, SPECIAL OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

ATTRIBUTION


THANK YOU


GPUOpen