GPU TECHNOLOGY CONFERENCE

April 4-7, 2016 | Silicon Valley

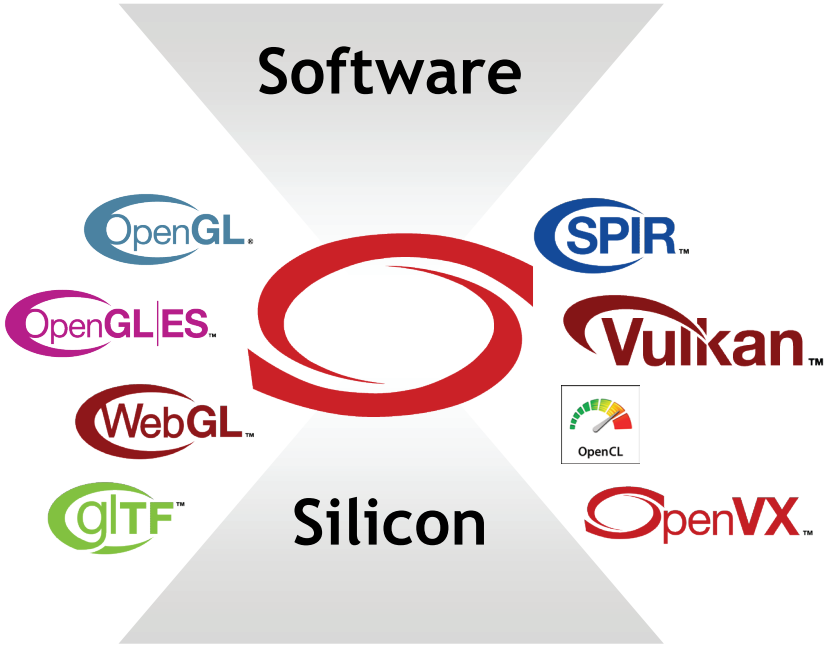
KHRONOS™ GROUP

Ecosystem Overview

Neil Trevett | Khronos President
NVIDIA Vice President Developer Ecosystem
ntrevett@nvidia.com | @neilt3d

PRESENTED BY NVIDIA.

Khronos Mission

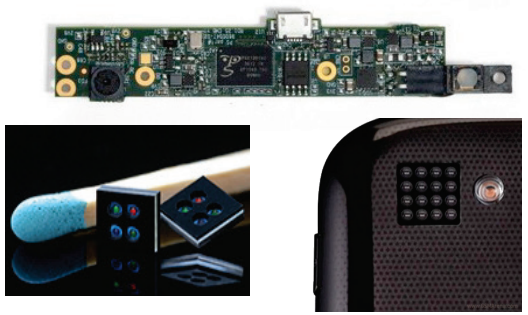


Khronos is an Industry Consortium of over 100 companies creating royalty-free, open standard APIs to enable software to access hardware acceleration for graphics, parallel compute and vision

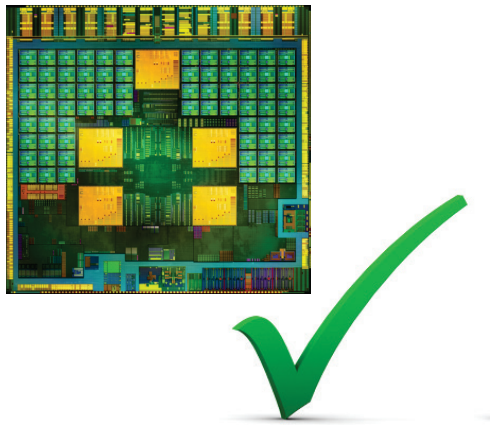
http://accelerateyourworld.org/

Vision Pipeline Challenges and Opportunities

Growing Camera Diversity



Diverse Vision Processors



Sensor Proliferation

Flexible sensor and camera control to GENERATE an image stream

OpenKCam™

Use efficient acceleration to PROCESS the image stream

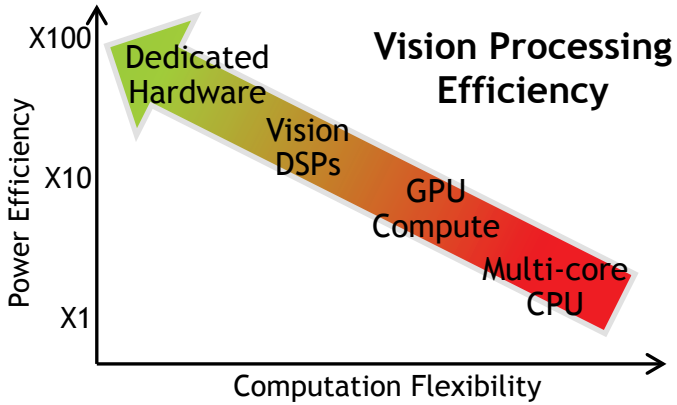
OpenVX™

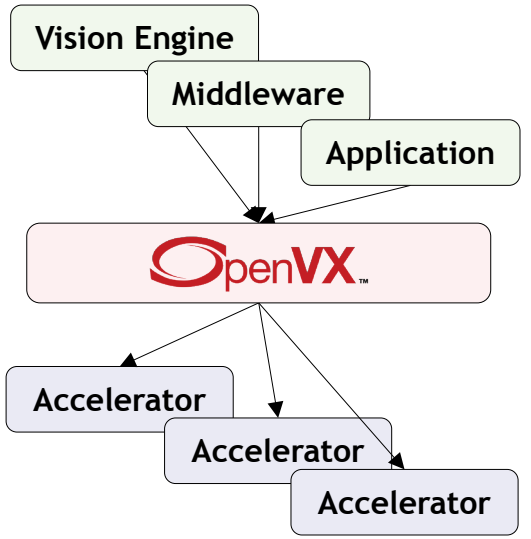
Combine vision output with other sensor data on device

StreamInput™

OpenVX – Low Power Vision Acceleration

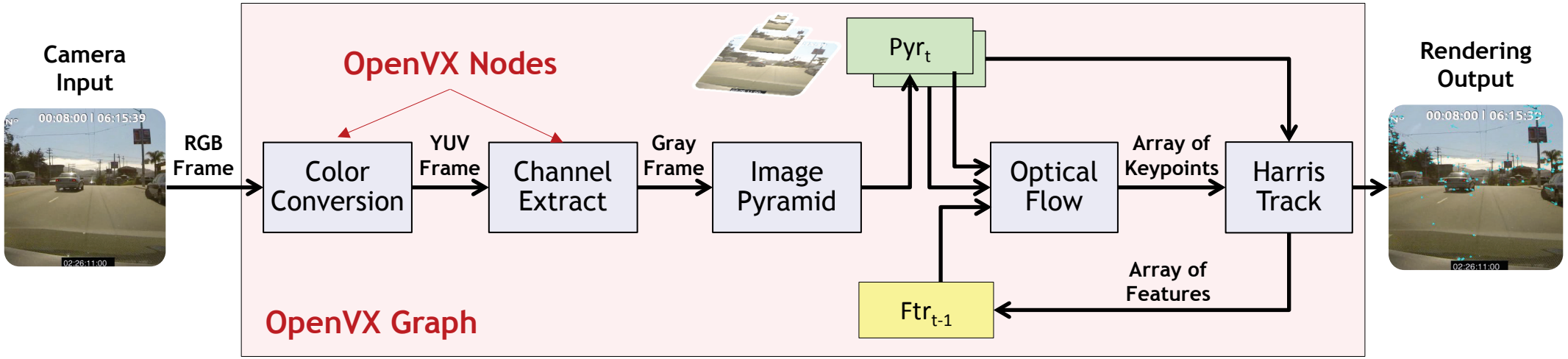
- **Higher level abstraction API**
  - Targeted at real-time mobile and embedded platforms
- **Performance portability across diverse architectures**
  - Multi-core CPUs, GPUs, DSPs and DSP arrays, ISPs, Dedicated hardware…
- **Extends portable vision acceleration to very low power domains**
  - Doesn't require high-power CPU/GPU Complex
  - Lower precision requirements than OpenCL
  - Low-power host can setup and manage frame-rate graph





OpenVX Graphs

- **OpenVX developers express a graph of image operations ('Nodes')**
  - Nodes can be on any hardware or processor coded in any language
  - E.g. on GPU nodes may be implemented in OpenCL or CUDA
- **Minimizes host interaction during frame-rate graph execution**
  - Host processor can setup graph which can then execute almost autonomously



Feature Extraction Example Graph

OpenVX Framework Efficiency..

Graph Scheduling

Split the graph execution across the whole system: CPU / GPU / dedicated HW

Faster execution or lower power consumption

Memory Management

Reuse pre-allocated memory for multiple intermediate data

Less allocation overhead, more memory for other applications

Kernel Merge

Replace a sub-graph with a single faster node

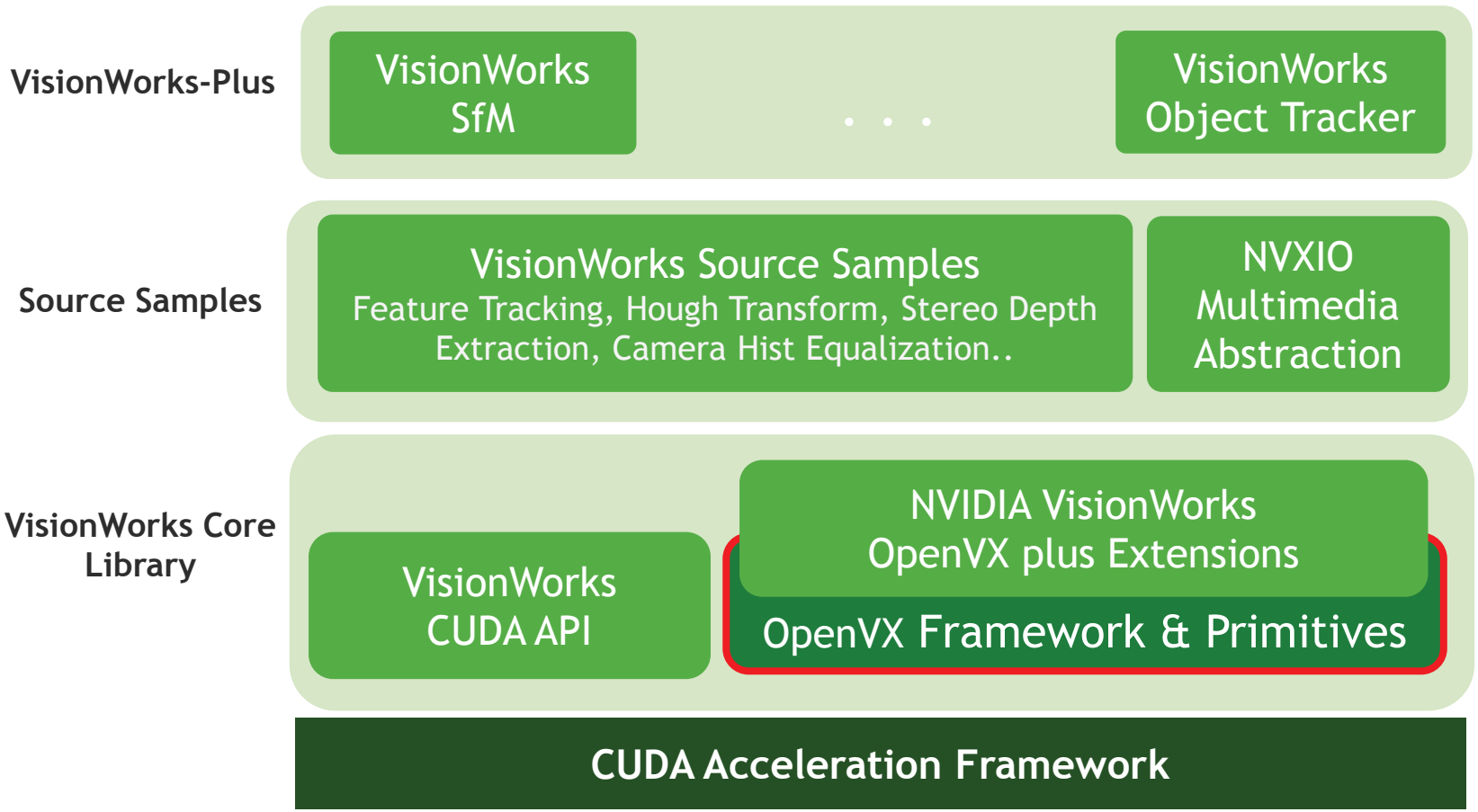
Better memory locality, less kernel launch overhead

Data Tiling

Execute a sub-graph at tile granularity instead of image granularity

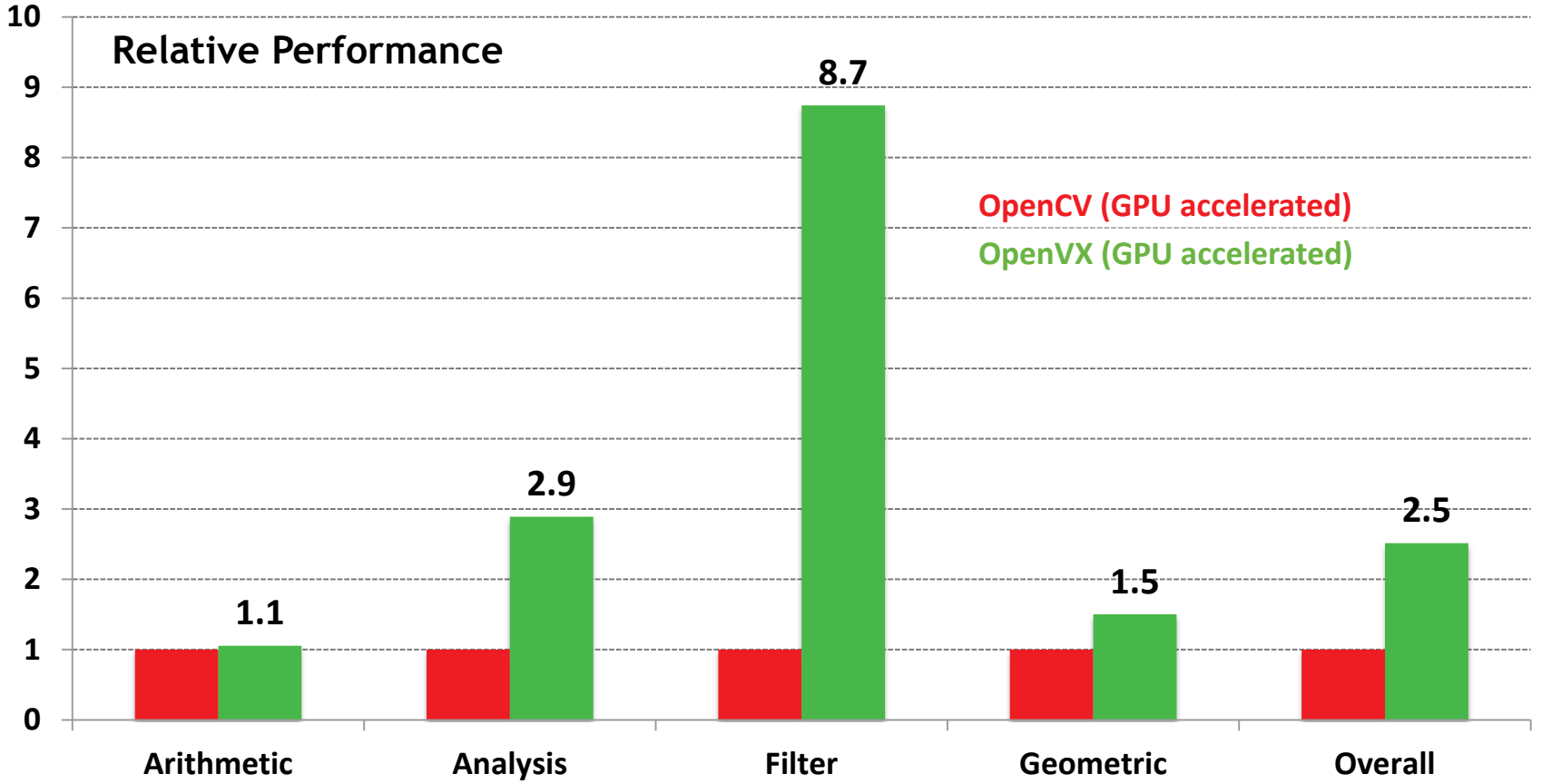
Better use of data cache and local memory

NVIDIA VisionWorks™ Toolkit Software Stack



Example Relative Performance

- **NVIDIA implementation experience**
  - Geometric mean of >2200 primitives, grouped into each categories, running at different image sizes and parameter settings



OpenVX Status

- **Finalized OpenVX 1.0 specification released October 2014**
  - OpenVX 1.0.1 spec maintenance update released June 2015 www.khronos.org/openvx
- **Khronos open source sample implementation of OpenVX 1.0 released**
  - https://www.khronos.org/registry/vx/sample/openvx_sample_20141217.tar.gz
- **Full conformance test suite and Adopters Program available**
  - Test suite exercises graph framework and functionality of each OpenVX 1.0 node
- **Commercial conformant products**
  - AMD, Intel, Imagination, NVIDIA, Synopsis, Vivante and many more coming…
- **Roadmap discussions**
  - More nodes, node profile sets, programmable nodes (SPIR-V?), Neural Net nodes

Layered Vision Processing Ecosystem

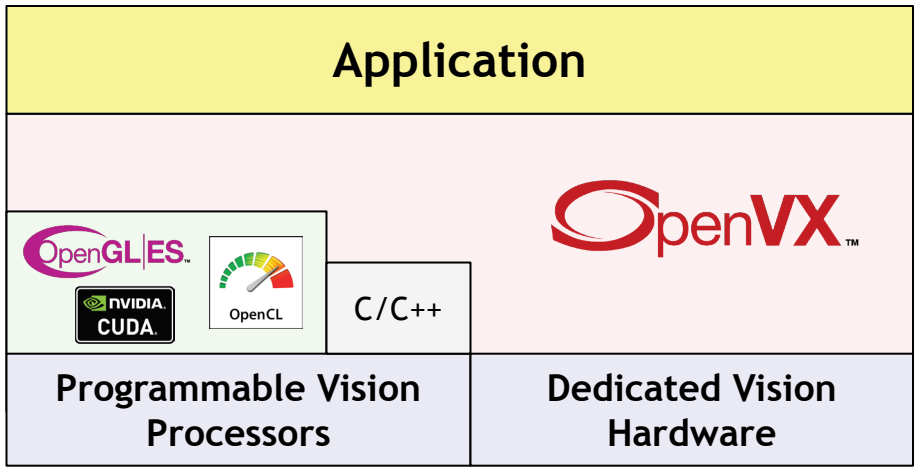
- **High-level graph abstraction gives implementation flexibility**
  - And significant optimization opportunities
- **Lower-level compute APIs *can* be used to *implement* OpenVX nodes**
  - Depending on the available processors
  - E.g. use OpenCL or OpenGL Compute Shaders if running on a GPU
- **Developers can then *use* OpenVX to easily connect those nodes into a graph**
  - With portable performance

Application Software

OpenVX Runtime

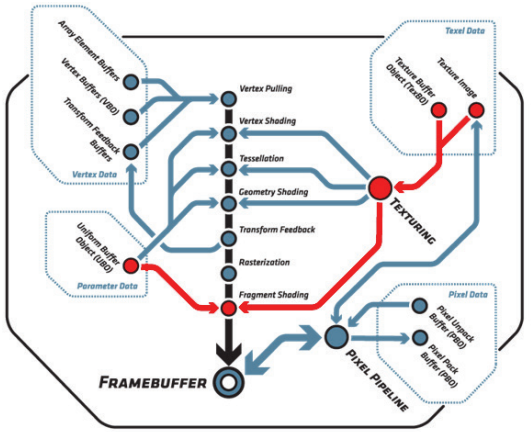
Flexible low-level APIs / languages

Processor Hardware



OpenGL ES Fragment Shaders

- **Fragment Shaders in OpenGL 2.0 were the original 'GPGPU' technique (2004)**
  - Fragment shaders executed as part of graphics pipeline
  - Need to configure inputs/outputs as textures and images
- **Mobile fragment shaders arrived in OpenGL ES 2.0 (2007)**
  - Now pervasively available on almost ANY mobile device or OS
- **Easy integration of compute shaders into graphics apps – no API interop needed**
  - Program kernels (shaders) in GLSL not full C
  - Good for small kernels NOT complete apps
  - Limited to acceleration on a single GPU

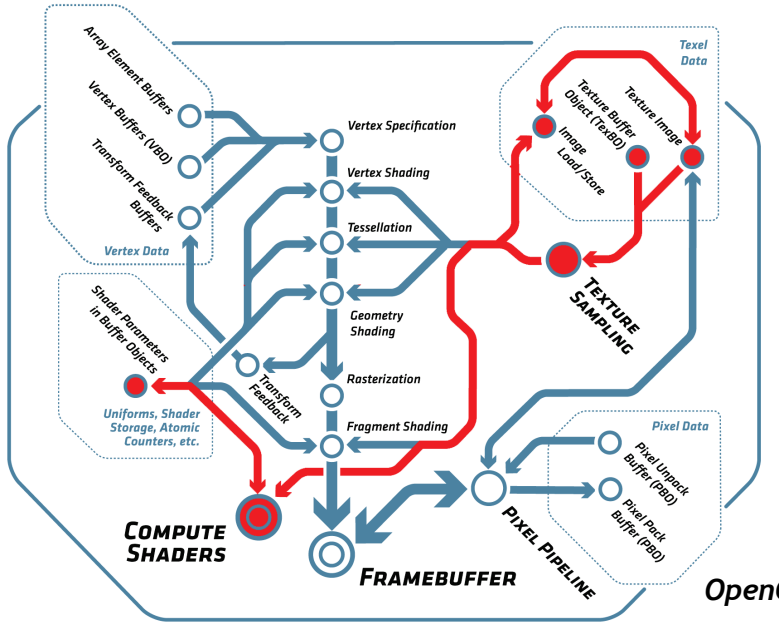


OpenGL 4.2 Pipeline

OpenGL ES	OpenGL ES Fragment Shaders
Governance	Khronos
Acceleration Devices	GPU
Scope	Graphics+Compute
Explicit Memory/Execution	No
Host Bindings	C
Kernel Language	GLSL
Availability	OpenGL ES 2.0+
Precision	NO IEEE 754 (highp) mediump / lowp

OpenGL ES Compute Shaders

- **New class of shader introduced in OpenGL 4.4 (2012) and OpenGL ES 3.1 (2014)**
  - Separated from graphics pipe – can use any buffer, image or texture
- **Much more flexibility on how compute shader is executed and uses memory**
  - But still use GLSL to write kernels and limited to execution on GPU only
- **OpenGL ES 3.1 is less pervasive than fragment shaders**
  - But mandated in Android Lollipop and Marshmallow – so growing rapidly

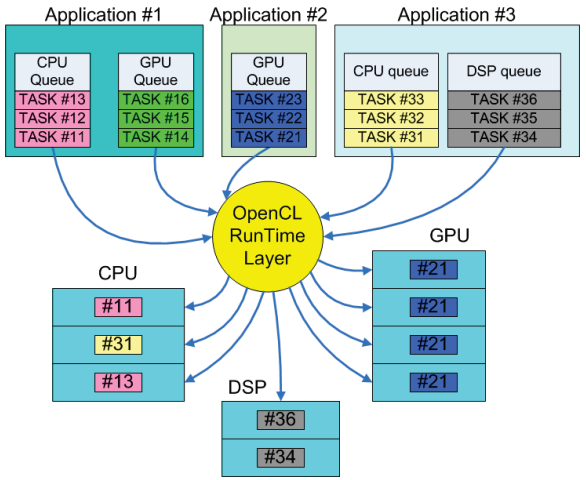


OpenGL 4.3 Pipeline

OpenGL ES	OpenGL ES Compute Shaders
Governance	Khronos
Acceleration Devices	GPU
Scope	Graphics+Compute
Explicit Memory/Execution	Yes
Host Bindings	C
Kernel Language	GLSL
Availability	OpenGL ES 3.1+
Precision	IEEE 754 Subset (highp) mediump / lowp

OpenCL

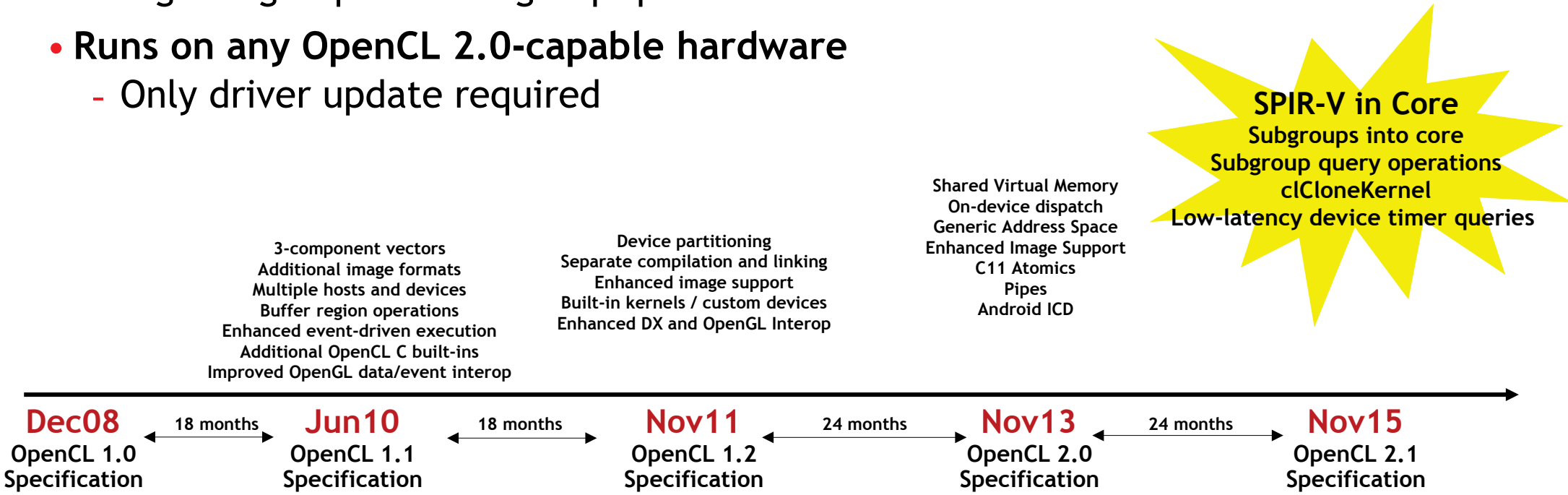
- **Heterogeneous parallel programming of diverse compute resources**
  - Targeting supercomputers -> mobile devices -> embedded systems
- **One code tree can be executed on CPUs, GPUs, DSPs, FPGA and hardware**
  - Distribute work across all available processors in a system
- **Can represent function of hardware 'Custom Devices' as built-in kernels**
  - Control hardware from OpenCL run-time: e.g. video encode/decode, Camera ISP
- **Robust framework for coding complete applications**
  - One source with both CPU *and* accelerated paths



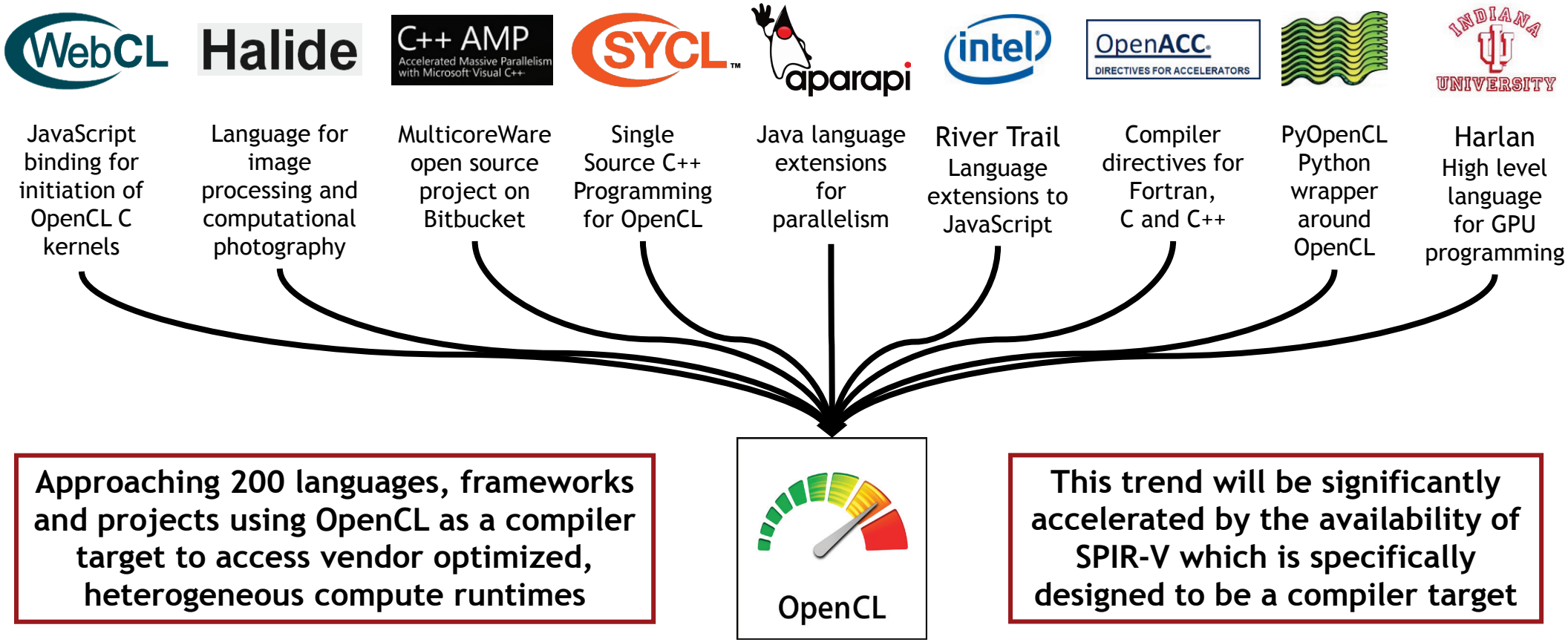
OpenCL	OpenCL
Governance	Khronos
Acceleration Devices	Heterogeneous
Scope	Compute
Explicit Memory/Execution	Yes
Host Bindings	C/C++
Kernel Language	C/C++/SPIR-V
Availability	Any OS
Precision	Full IEEE 754 IEEE 754 Subset IEEE 754 Relaxed

OpenCL 2.1 Released - November 2015

- **Support for the SPIR-V 1.0 intermediate language in core**
  - E.g. SPIR-V used to ingest from diverse language front-ends
  - OpenCL C ingestion still supported to preserve kernel code investment
- **OpenCL API updates**
  - E.g. subgroups and subgroup queries in core
- **Runs on any OpenCL 2.0-capable hardware**
  - Only driver update required



OpenCL as Parallel Language Backend



Approaching 200 languages, frameworks and projects using OpenCL as a compiler target to access vendor optimized, heterogeneous compute runtimes

This trend will be significantly accelerated by the availability of SPIR-V which is specifically designed to be a compiler target

SPIR-V Transforms the Language Ecosystem

- **First multi-API, intermediate language for parallel compute and graphics**
  - Native representation for Vulkan shader and OpenCL kernel source languages
  - https://www.khronos.org/registry/spir-v/papers/WhitePaper.pdf
- **Cross vendor intermediate representation**
  - Language front-ends can easily access multiple hardware run-times
  - Acceleration hardware can leverage multiple language front-ends
  - Encourages tools for program analysis and optimization in SPIR form

Multiple Developer Advantages

Same front-end compiler for multiple platforms
Reduces runtime kernel compilation time
Don't have to ship shader/kernel source code
Drivers are simpler and more reliable

Evolution of SPIR Family

- **SPIR-V is first fully specified Khronos-defined SPIR standard**
  - Does not use LLVM to isolate from LLVM roadmap changes
  - Includes full flow control, graphics and parallel constructs beyond LLVM
  - Khronos will open source SPIR-V <-> LLVM conversion tools

SPIR	SPIR 1.2	SPIR 2.0	SPIR-V 1.0
LLVM Interaction	Uses LLVM 3.2	Uses LLVM 3.4	100% Khronos defined Round-trip lossless conversion
Compute Constructs	Metadata/Intrinsics	Metadata/Intrinsics	Native
Graphics Constructs	No	No	Native
Supported Language Feature Set	OpenCL C 1.2	OpenCL C 1.2 OpenCL C 2.0	OpenCL C 1.2 / 2.0 OpenCL C++ GLSL
OpenCL Ingestion	OpenCL 1.2 Extension	OpenCL 2.0 Extension	OpenCL 2.1 Core
Vulkan Ingestion	-	-	Vulkan Core

Driving the SPIR-V Open Source Ecosystem

Khronos has open sourced these tools and translators

Khronos plans to open source these tools soon

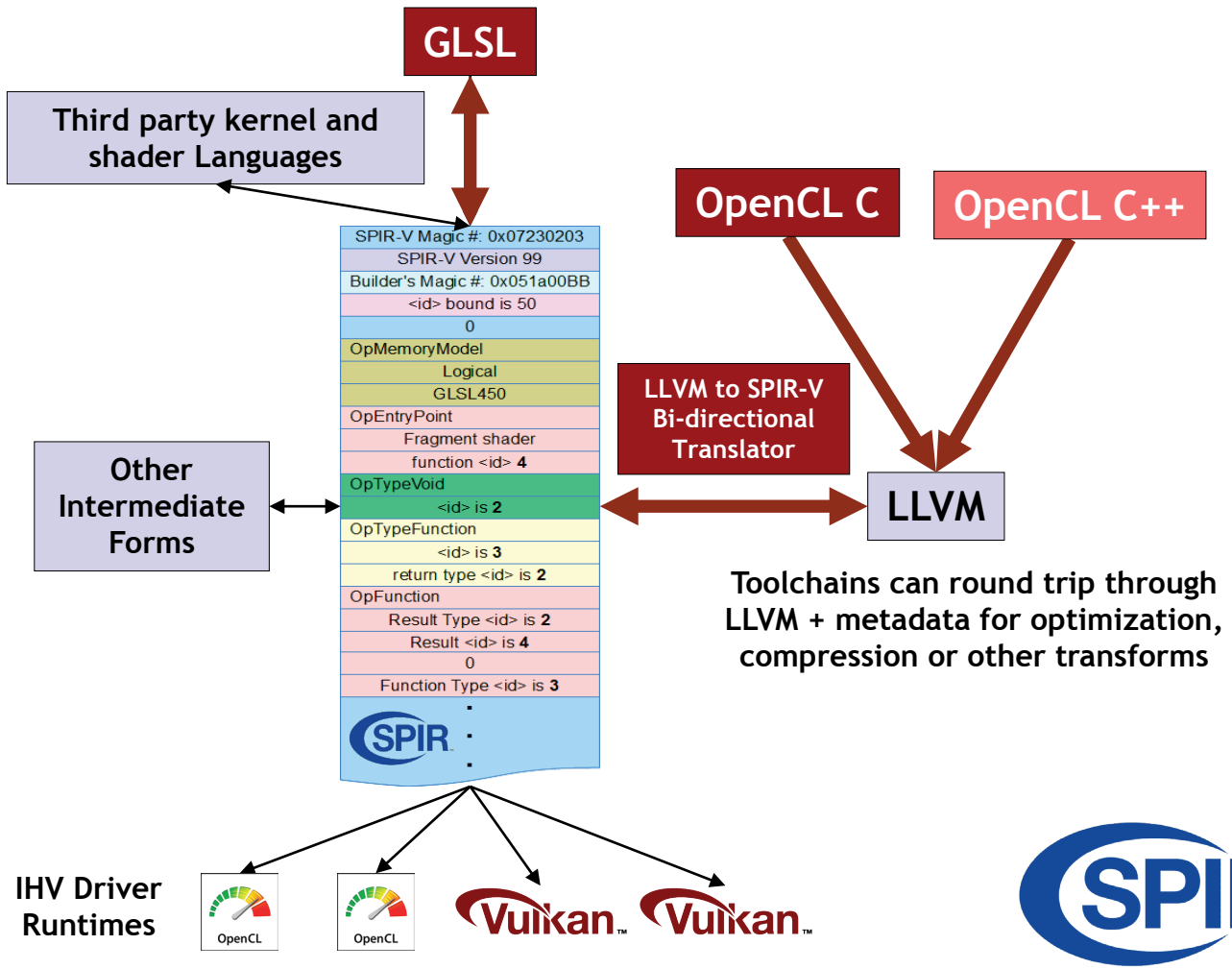
SPIR-V Tools

SPIR-V Validator

SPIR-V (Dis)Assembler

SPIR-V

- 32-bit Word Stream
- Extensible and easily parsed
- Retains data object and control flow information for effective code generation and translation



The Need for a New Generation GPU API

- **Explicit**
  - Open up the high-level driver abstraction to give direct, low-level GPU control
- **Streamlined**
  - Faster performance, lower overhead, less latency
- **Portable**
  - Cloud, desktop, console, mobile and embedded
- **Extensible**
  - Platform for rapid innovation

OpenGL has evolved over 25 years and continues to meet industry needs - but there is a need for a complementary API approach

GPUs are increasingly programmable and compute capable + platforms are becoming mobile, memory-unified and multi-core

GPUs will accelerate graphics, compute, vision and deep learning across diverse platforms: FLEXIBILITY and PORTABILITY are key

New Generation GPU APIs

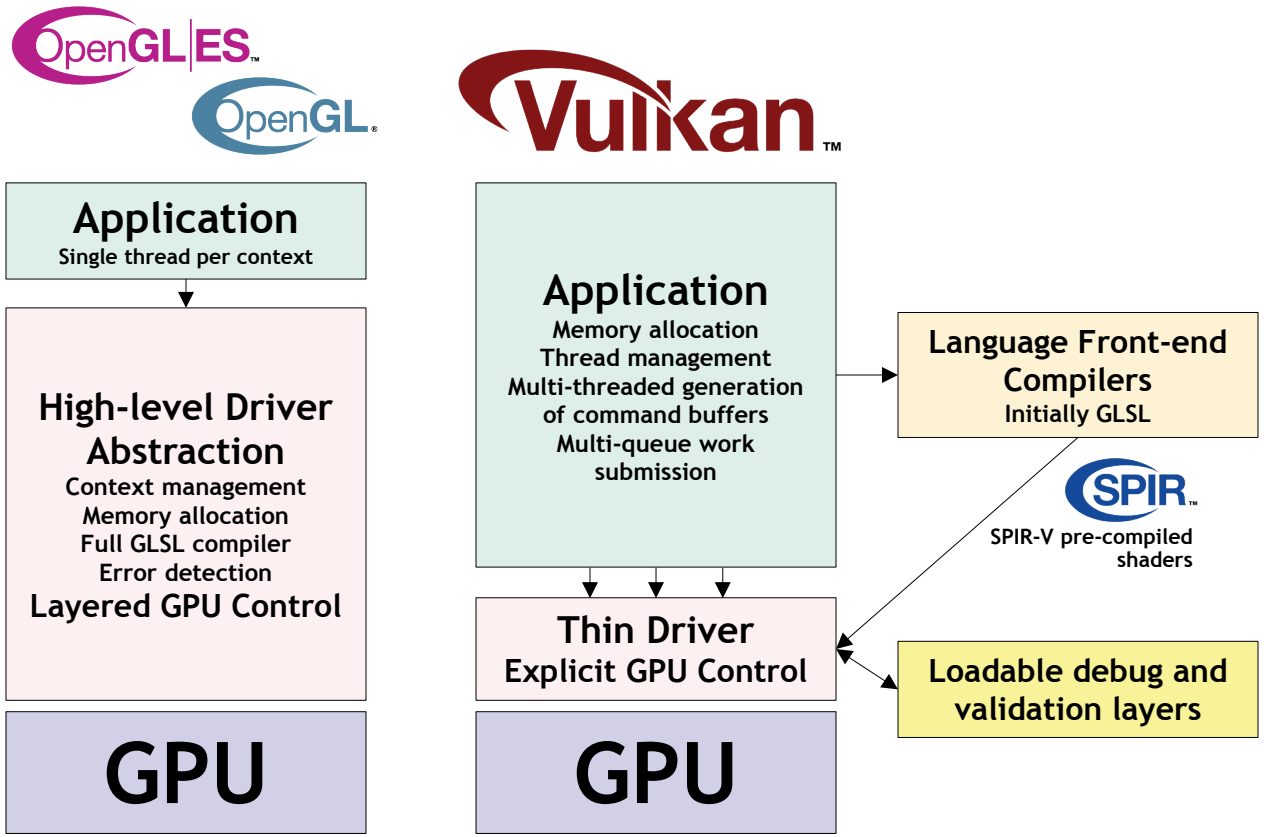
Only Windows 10

Only Apple

Vulkan™

Cross Platform

Vulkan Explicit GPU Control



Vulkan 1.0 provides access to
OpenGL ES 3.1 / OpenGL 4.X-class GPU functionality
but with increased performance and flexibility

Vulkan Benefits

Simpler drivers:
Improved efficiency/performance
Reduced CPU bottlenecks
Lower latency
Increased portability

Resource management in app code:
Less hitches and surprises

Command Buffers:
Command creation can be multi-threaded
Multiple CPU cores increase performance

Graphics, compute and DMA queues:
Work dispatch flexibility

SPIR-V Pre-compiled Shaders:
No front-end compiler in driver
Future shading language flexibility

Loadable Layers
No error handling overhead in production code

Vulkan - No Compromise Performance

Vulkan Genesis

Khronos members from all segments of the graphics industry agree the need for new generation cross-platform GPU API

Significant proposals, IP contributions and engineering effort from many working group members

Vulkan

Khronos' first API 'hard launch' 16Feb16

18 months

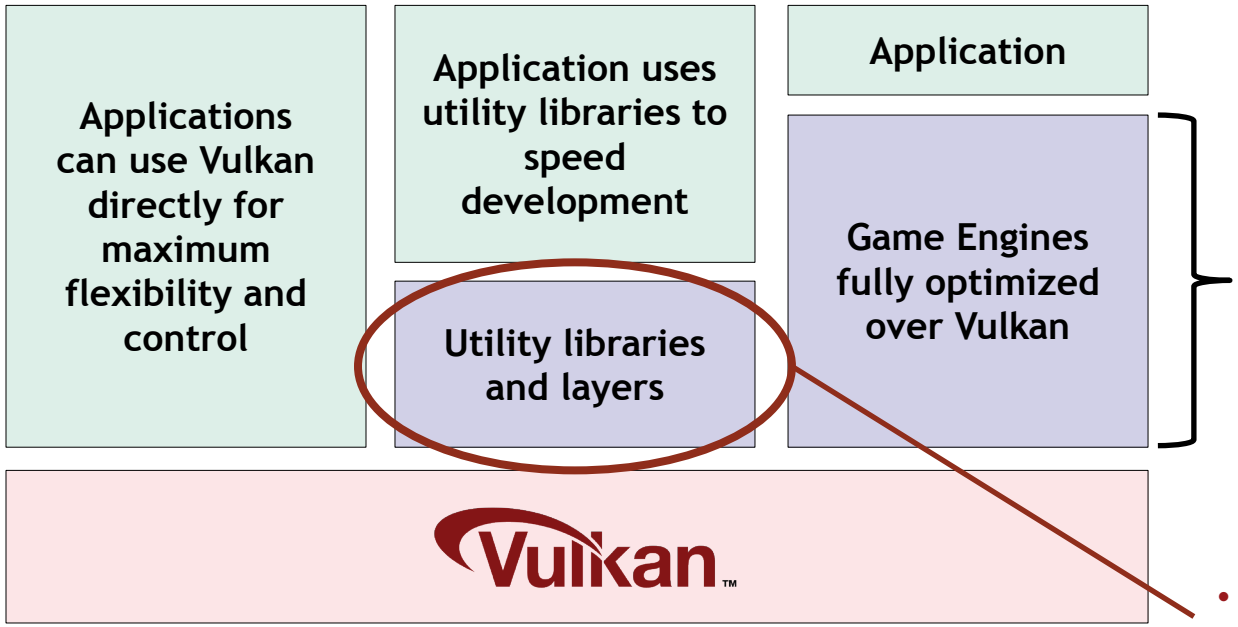
Including an unprecedented level of participation from game engine developers

A high-energy working group effort

Specification, Conformance Tests, SDKs - all open source…
Reference Materials, Compiler front-ends, Samples…
Multiple Conformant Drivers on multiple OS

Vulkan Working Group Participants

The Power of a Three Layer Ecosystem



Applications using game engines will automatically benefit from Vulkan's enhanced performance

OXIDE · LUCASFILM Ltd · BLIZZARD ENTERTAINMENT · EA · unity · EPIC GAMES · VALVE

Rich Area for Innovation

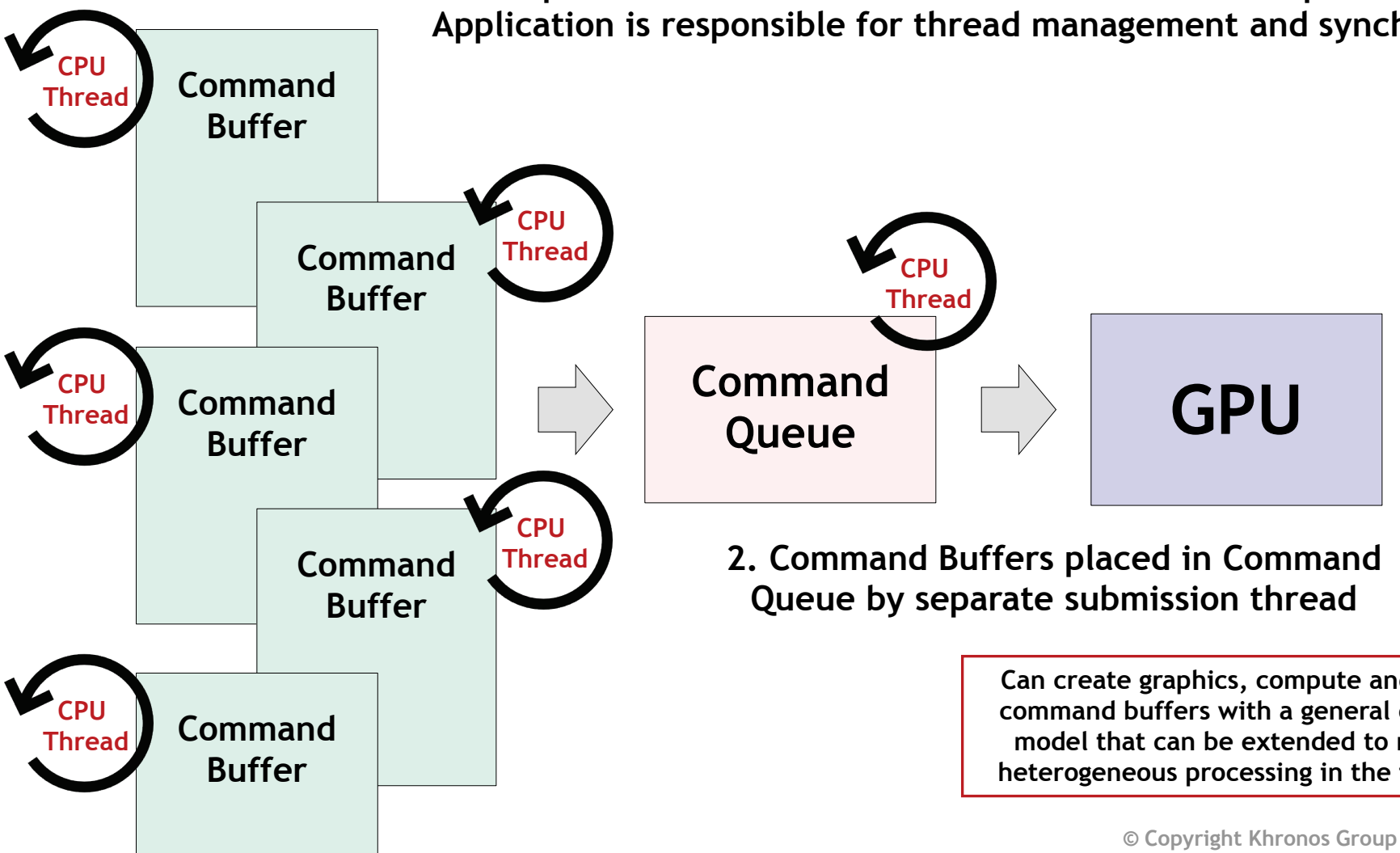
- Many utilities and layers will be in open source
- Layers to ease transition from OpenGL
- Domain specific flexibility

Similar ecosystem dynamic as WebGL

A widely pervasive, powerful, flexible foundation layer enables diverse middleware tools and libraries

Vulkan Multi-threading Efficiency

1. Multiple threads can construct Command Buffers in parallel
Application is responsible for thread management and synch

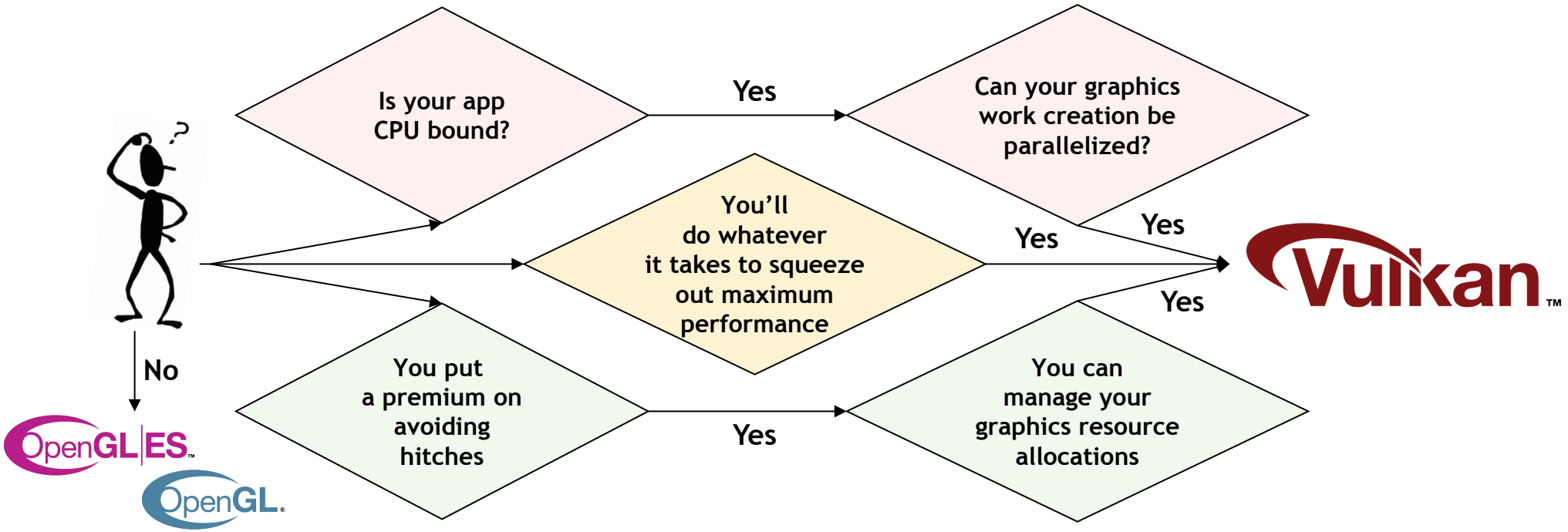


2. Command Buffers placed in Command Queue by separate submission thread

Can create graphics, compute and DMA command buffers with a general queue model that can be extended to more heterogeneous processing in the future

Which Developers Should Use Vulkan?

- **Vulkan puts more work and responsibility into the application**
  - Not every developer will need or want to make that extra investment
- **For many developers OpenGL and OpenGL ES will remain the most effective API**
  - Khronos actively evolving OpenGL and OpenGL ES in parallel with Vulkan



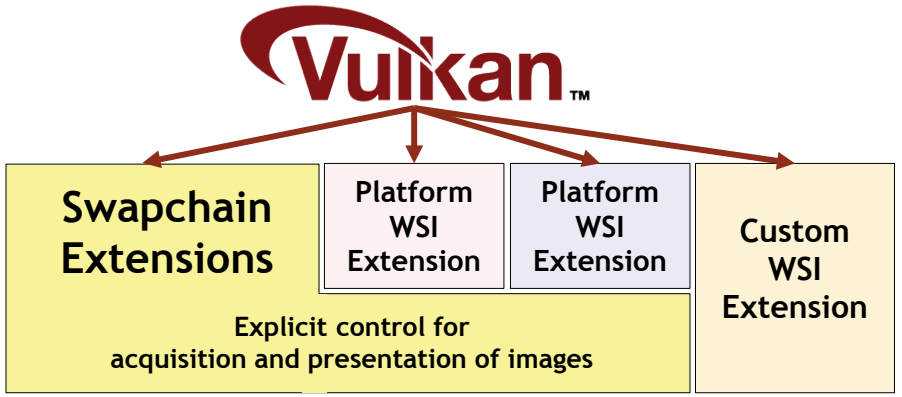
Vulkan provides more choice to developers and can be used to create new classes of end-user experience

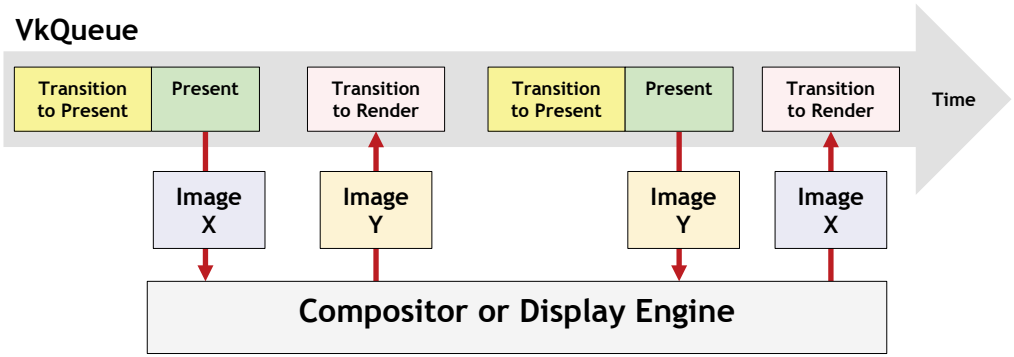
Vulkan Feature Sets

- **Vulkan supports hardware with a wide range of hardware capabilities**
  - Mobile OpenGL ES 3.1 up to desktop OpenGL 4.5 and beyond
- **One unified API framework for desktop, mobile, console, and embedded**
  - No "Vulkan ES" or "Vulkan Desktop"
- **Vulkan precisely defines a set of "fine-grained features"**
  - Features are specifically enabled at device creation time (similar to extensions)
- **Platform owners define a Feature Set for their platform**
  - Vulkan provides the mechanism but does not mandate policy
  - Khronos will define Feature Sets for platforms where owner is not engaged
- **Khronos will define feature sets for Windows and Linux**
  - After initial developer feedback

Vulkan Window System Integration (WSI)

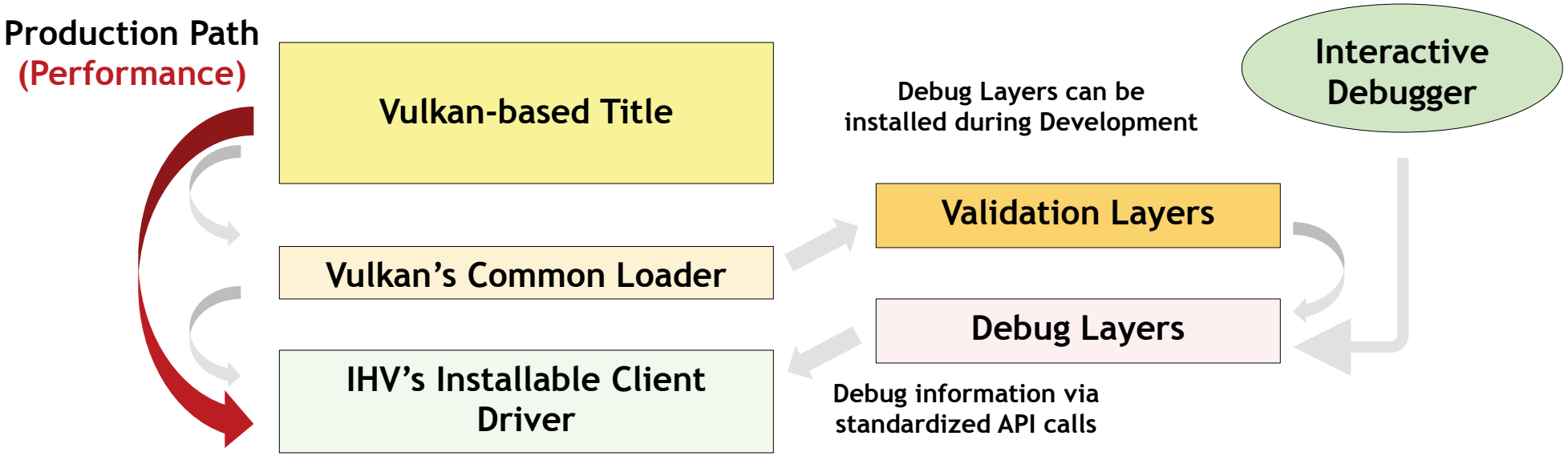
- **Explicit control for acquisition and presentation of images**
  - Designed to fit the Vulkan API and today's compositing window systems
  - Cleanly separates device creation from window system
- **Platform provides an array of persistent presentable images = Vulkan Swapchain**
  - Device exposes which queues support presentation
  - Application explicitly controls which image to render and present
- **Standardized extensions - unified API for multiple window systems**
  - Works across Android, Mir, Windows (Vista and up), Wayland and X (with DRI3)
  - Platforms can extend functionality, define custom WSI stack, or have no display at all



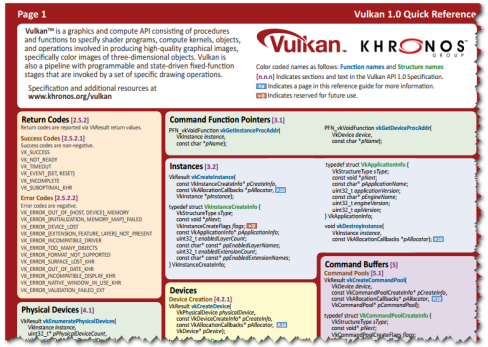


Vulkan Tools Architecture

- **Layered design for cross-vendor tools innovation and flexibility**
  - IHVs plug into a common, extensible architecture for code validation, debugging and profiling during development without impacting production performance
- **Khronos Open Source Loader enables use of tools layers during debug**
  - Finds and loads drivers, dispatches API calls to correct driver and layers



Vulkan Developer Resources



KHRONOS GROUP

GET INVOLVED! HELP US EVOLVE THE VULKAN ECOSYSTEM

Khronos has placed an unprecedented amount of materials into open source so you can provide feedback, showcase your work, fix bugs, and extend Vulkan capabilities for the future. Get engaged AND show the world what YOU are doing with Vulkan.

Vulkan resources on Github

Discussions

Issue Trackers:

- **Vulkan Specification, Reference Pages, and API Registry**
- **Vulkan CTS**
- **Vulkan Loader and Validation Tools**
- **Vulkan Sample Code**
- **Data Format specification**

Khronos.org Canonical Resources

Specifications, Header Files
Feature Set Definitions
(Windows and Linux - post developer feedback)
Quick Reference and Reference Pages
Conformance Test Source and Test Process

Materials to Build SDKs and Tools

Compiler toolchain sources
Validation Layer Source
Loader Source
Layers and Loader documentation
(open source resources in github.com/KhronosGroup)

Everything needed to create SDKs for any platform or market

LunarG

Windows and Linux Installable SDKs
Loader and Validation Layer binaries
Tools Layers - source and binaries
Samples - source and binaries
Windows get started guide

IHV Websites

Drivers and Loader
Vendor tools and layers

Third Party Websites

Layers, Samples etc.

VULKAN DRIVERS

Behind every great API are the drivers that bring it life on your GPU. Download the latest drivers for your system that now include Vulkan 1.0.

AMD Imagination intel NVIDIA QUALCOMM

DEMOS, SAMPLES & ENGINES

Download demos and open source samples to take your new Vulkan API for a test drive – and get a heads up on Vulkan resources which will be arriving soon…

ARM CINDER Imagination intel LUNARG MoltenVK Norbert Nopper NVIDIA Sascha Willems UNREAL ENGINE

Vulkan Ecosystem – Launch + 6 Weeks

http://www.pcworld.com/article/3035020/linux/valves-steamos-now-supports-vulkan-the-cross-platform-alternative-to-directx-12.html

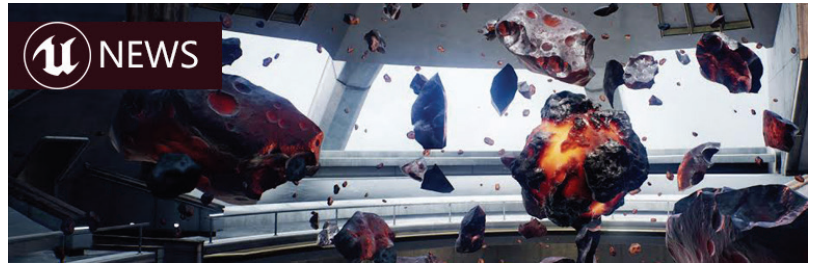
Valve porting Source 2 Engine to Vulkan

MetalGL

MetalVK

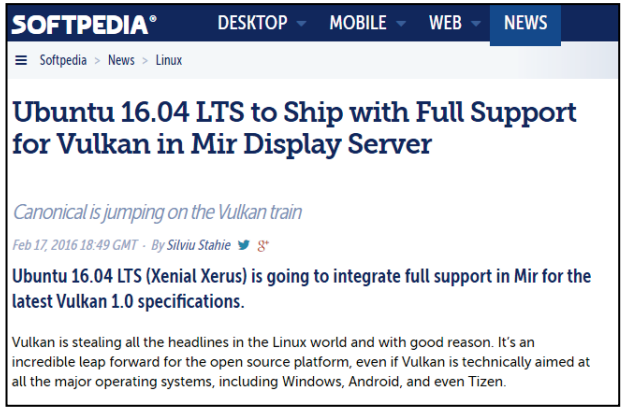
Vulkan and OpenGL ES over Metal - in development By Brenwill Workshop

https://www.youtube.com/watch?v=FnKu7MLB7vQ



Unreal Engine 4 ported to Vulkan. 'ProtoStar' demo shown at MWC 2016

CroTeam shipped a beta Vulkan back-end to Talos Principle on the same day that the specification was launched



http://news.softpedia.com/news/ubuntu-16-04-lts-to-ship-with-full-support-for-vulkan-in-mir-display-server-500543.shtml

LunarG Vulkan SDK shipped in open source on launch day

Vulkan Benchmarks Coming Soon

Kishonti adding Vulkan support to GFXBench 5.0

Basemark GPU Vulkan - commercially available in Q2, 2016 for both mobile and PC platforms

Futuremark will release a new version of the Slingshot benchmark that supports Vulkan

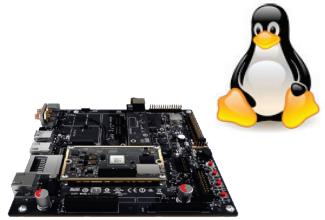
Vulkan and NVIDIA

https://developer.nvidia.com/Vulkan

WHQL and Vulkan Conformant Game Ready drivers on Windows PCs with GeForce and Quadro

Vulkan Conformant Drivers on GeForce and Quadro on Linux PCs

Vulkan Conformant Drivers on Shield Android TV and developer drivers on Shield Tablet

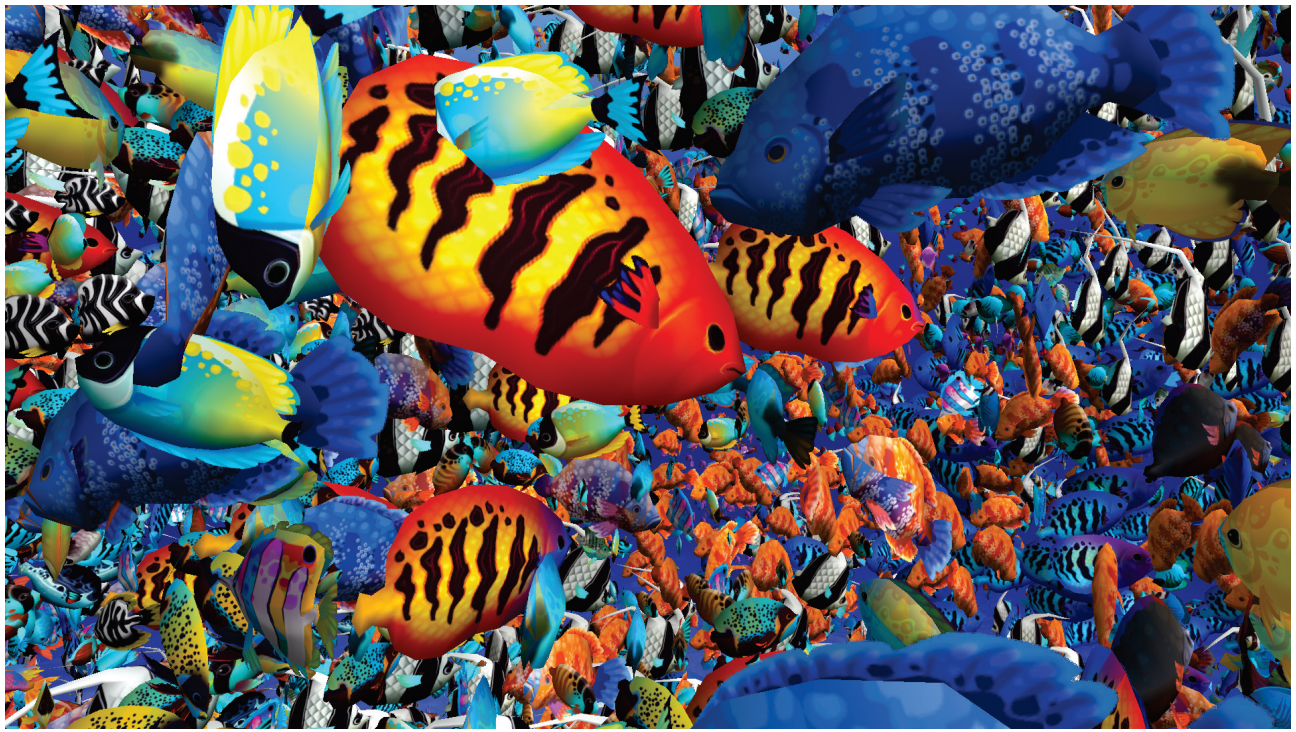


Vulkan Conformant Drivers on Jetson TX1 with Embedded Linux

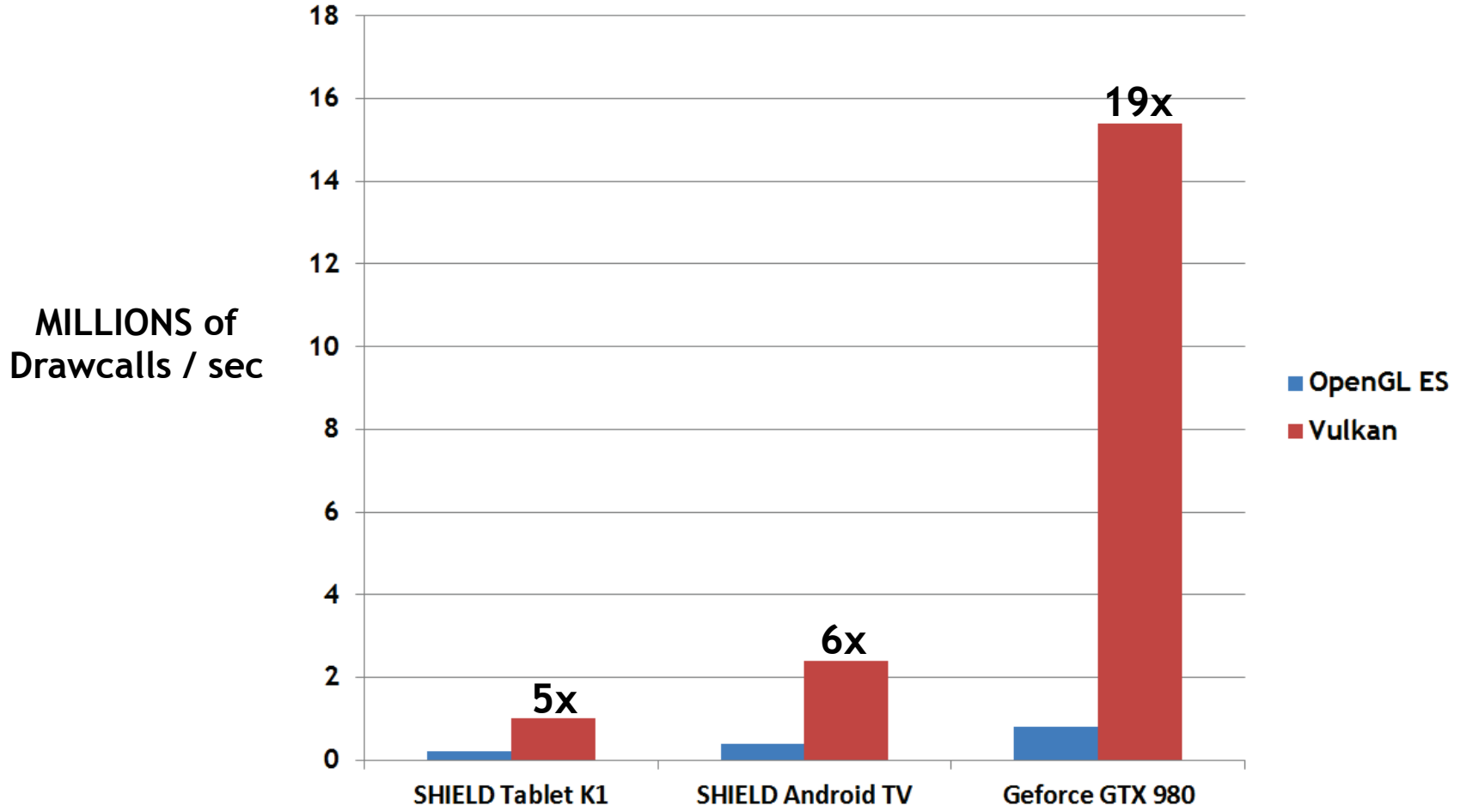
> We have been using NVIDIA hardware and drivers on both Windows and Android for Vulkan development, and the reductions in CPU overhead have been impressive.
>
> — John Carmack, Chief Technology Officer, Oculus

Vulkan Demo – Fish!

- **Compare performance between Vulkan and OpenGL ES 3.1**
  - Worker threads create command buffers in Vulkan mode
- **Runs on Android and Windows**
  - Source code will be available soon



200K Fish, 1 Fish / Drawcall



Safety Critical Working Group

OpenGL SC™
2005
OpenGL SC 1.0
Fixed function graphics subset

OpenGL SC™
2016 (planned)
OpenGL SC 2.0
Programmable shader pipeline subset

OpenGL ES™
2003
OpenGL ES 1.0
Fixed function graphics

OpenGL ES™
2007
OpenGL ES 2/3
Programmable shader pipeline

OpenVX™

Vulkan™

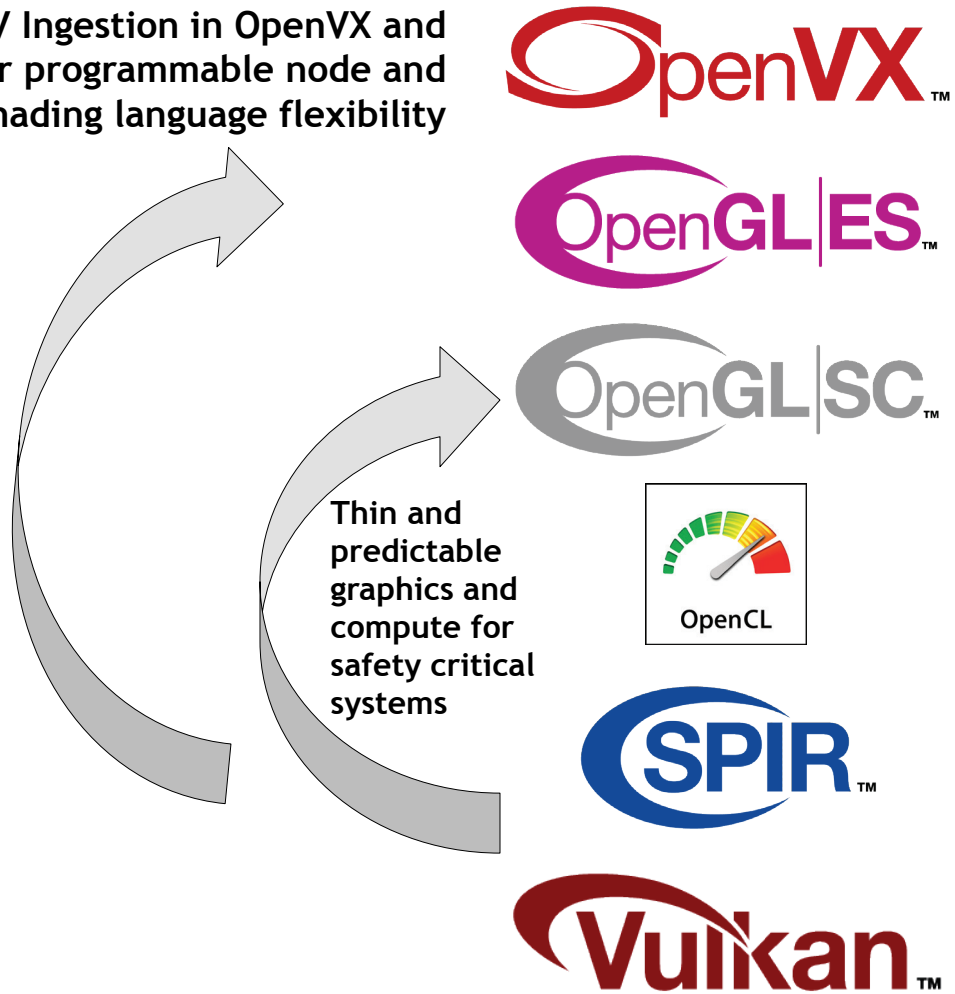
New Generation API for safety certifiable graphics AND compute

Many future safety critical use cases involve vision and compute acceleration (e.g. neural nets)

New Khronos Safety Critical Advisory Panel
Defining guidelines for creating specifications for ISO 26262 and DO-178B/C certification

Roadmap Possibilities in Discussion

SPIR-V Ingestion in OpenVX and OpenGL ES for programmable node and shading language flexibility



Thin and predictable graphics and compute for safety critical systems

Khronos members decide how to evolve and mix and match a rich set of APIs and technologies to meet market needs

1. C++ Shading Language
2. Single source C++ Programming from SYCL
3. OpenCL-class Heterogeneous Compute to Vulkan runtime

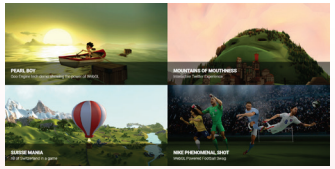
Khronos Open Standards for Graphics and Compute

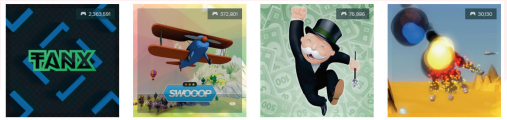
			LATEST STATUS
1990's	Workhorse cross-platform professional 3D apps & gaming	OpenGL	New Extensions to enable latest desktop graphics capabilities
2000's	Ubiquitous mobile gaming & graphics apps	OpenGL ES	OpenGL ES 3.2 released to bring AEP functionality to core
2005	Safety Critical Graphics	OpenGL SC	New Safety Critical Working Group - Call for Participation
2008	Heterogeneous parallel compute	OpenCL	OpenCL 2.1 specification and C++ Headers released
2014	Portable intermediate representation for graphics and parallel compute	SPIR	SPIR-V 1.0 released and significant open source activity
2016	High-efficiency GPU graphics and compute for performance critical apps	Vulkan	Released in Feb16. Adopted by Android. Building ecosystem

Vulkan at GTC

What	When
Vulkan and NVIDIA: The Essentials	Monday 9AM
High-Performance, Low-Overhead Rendering with OpenGL and Vulkan	Monday 10AM
Khronos Ecosystem Overview	Monday 11AM
VKCPP: A C++ Layer on Top of Vulkan	Monday 1PM
Hangout: Vulkan	Monday 2PM
Robust Software Development: Bug Prevention and Isolation	Monday 3PM
GPU-Driven Rendering in Vulkan and OpenGL	Monday 4PM
Hangout: Vulkan C++ API	Tuesday 1PM
Hangout: Vulkan	Tuesday 3PM

WebGL, WebVR and glTF Meetup





>1 Billion desktop and mobile browsers
Including Facebook gaming

WebGL™

sceneJS
3D Engine for the Web

unity 5

blend4web

CESIUM

PEX

xeoEngine

babylon JS

CopperLicht

PLAYCANVAS

three.js

Audio	Video	Images	3D
MP3	H.264	JPEG	glTF™
napster	YouTube™	facebook	!

glTF - OpenGL Transmission Format

Efficient transmission of 3D assets

A widely adopted media format ignites previously untapped commercials opportunities

WebGL Meetup
Wednesday 7-9PM Room 210E
Fast-paced WebGL, WebVR and glTF presentations and demos

Thank You!

- **NVIDIA information**
  - https://developer.nvidia.com/Vulkan
  - https://developer.nvidia.com/embedded/visionworks
- **Any company or organization is welcome to join Khronos for a voice and a vote in any of these standards**
  - www.khronos.org
- **Neil Trevett**
  - ntrevett@nvidia.com
  - @neilt3d

KHRONOS™ GROUP