Vulkan Overview

June 2015
Khronos Connects Software to Silicon

Open Consortium creating OPEN STANDARD APIs for hardware acceleration
Any company is welcome - many international members - one company one vote

Software

- ROYALTY-FREE specifications
- State-of-the art IP framework protects members AND the standards

Silicon

- Low-level silicon APIs needed on every platform
- Graphics, parallel compute, rich media, vision, sensor and camera processing

- International, non-profit organization
- Membership fees cover operating and engineering expenses

- API Specifications AND Conformance
- Tests for cross-vendor portability

Strong industry momentum
100s of man years invested by industry experts

Well over a BILLION people use Khronos APIs Every Day…
Access to 3D on Over 2 BILLION Devices

300M Desktops / year
Windows, Mac, Linux

1.9B Mobiles / year
Android, Apple

1B Browsers / year

Worldwide Device Shipments by Segment (Thousands of Units)

<table>
<thead>
<tr>
<th>Device Type</th>
<th>2012</th>
<th>2013</th>
<th>2014</th>
<th>2015</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC (Desk-Based and Notebook)</td>
<td>341,273</td>
<td>299,342</td>
<td>277,939</td>
<td>268,491</td>
</tr>
<tr>
<td>Tablet (Ultramobile)</td>
<td>119,529</td>
<td>179,531</td>
<td>263,450</td>
<td>324,565</td>
</tr>
<tr>
<td>Mobile Phone</td>
<td>1,746,177</td>
<td>1,804,234</td>
<td>1,993,425</td>
<td>1,964,788</td>
</tr>
<tr>
<td>Other Ultramobiles (Hybrid and Clamshell)</td>
<td>9,344</td>
<td>17,195</td>
<td>39,636</td>
<td>63,835</td>
</tr>
<tr>
<td>Total</td>
<td>2,216,322</td>
<td>2,300,402</td>
<td>2,474,451</td>
<td>2,621,678</td>
</tr>
</tbody>
</table>

Source: Gartner (December 2013)
The Need for Vulkan

In the twenty two years since OpenGL was invented - the architecture of GPUs and platforms has changed radically

GPUs being used for graphics, compute and vision processing on a rapidly increasing diversity of platforms - increasing the need for cross-platform standards

Ground-up design of a modern open standard API for driving high-efficiency graphics and compute on GPUs used across diverse devices

• Simpler drivers for low-overhead efficiency and cross vendor consistency
• Unified API for mobile, desktop, console and embedded platforms
• Layered architecture so validation and debug layers unloaded when not needed
Next Generation GPU APIs

Ground-up API designs for driving high-efficiency graphics and compute on GPUs

One OS

One Platform Vendor

All Modern Platforms and GPUs

An explicit API that is also cross-platform needs careful design balance. Unified memory and tiler architectures are first-class citizens. Vulkan-supported platforms will include Linux, Windows XP to Windows 10, Mobile.
Vulkan Explicit GPU Control

Complex drivers lead to driver overhead and cross vendor unpredictability

Error management is always active

Driver processes full shading language source

Separate APIs for desktop and mobile markets

Simpler drivers for low-overhead efficiency and cross vendor consistency

Layered architecture so validation and debug layers can be unloaded when not needed

Run-time only has to ingest SPIR-V intermediate language

Unified API for mobile, desktop, console and embedded platforms

Vulkan delivers the maximized performance and cross platform portability needed by sophisticated engines, middleware and apps
Vulkan Layered Ecosystem

Applications can use Vulkan directly for maximum flexibility and control

Applications use utility libraries to speed development

Utility libraries and layers

Application

Games Engines fully optimized over Vulkan

The industry’s leading games and engine vendors are participating in the Vulkan working group

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

Developers can choose at which level to use the Vulkan Ecosystem
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.
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

- Common Loader used to enable use of tools layers during debug
  - Cross-vendor API calls provide debug data

Production Path
(Performance)

Vulkan-based Title

Vulkan’s Common Loader

IHV’s Installable Client Driver

Debug Layers can be installed during Development

Validation Layers

Debug information via standardized API calls

Interactive Debugger
Vulkan Tools Ecosystem

• Extensible modular architecture encourages many fine-grained layers - new layers can be added easily

• Khronos encouraging open community of tools e.g. shader debugging

• Valve, LunarG, Codeplay and others are already driving the development of open source Vulkan tools

• Customized interactive debugging and validation layers will be available together with first drivers

Prototype Vulkan Debugger from Valve and LunarG
LunarG.com/Vulkan
Vulkan - Enhancing Driver Reliability

Streamlined API is easier to implement and test

Cross-vendor Portability

SPIR-V intermediate language improves shader program portability and reduces driver complexity

Open source conformance test source components for community engagement
Vulkan Language Ecosystem

GLSL Shader Source
- GLSL will be first shading language supported by Vulkan

GLSL to SPIR-V Translator

Game Engines
- Can flexibly target SPIR-V and Vulkan back-ends

Future diversity in domain-specific languages, frameworks and tools
- E.g. C++ Shading Language

SPIR-V ingest supported in Vulkan core

Vulkan Runtime

Device X  Device Y  Device Z

Khronos is considering open sourcing compiler front-ends

GLSL will be first shading language supported by Vulkan
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

• 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

SPIR-V is a significant convergence point in the language ecosystem for graphics and parallel computation
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

<table>
<thead>
<tr>
<th></th>
<th>SPIR 1.2</th>
<th>SPIR 2.0</th>
<th>SPIR-V</th>
</tr>
</thead>
<tbody>
<tr>
<td>LLVM Interaction</td>
<td>Uses LLVM 3.2</td>
<td>Uses LLVM 3.4</td>
<td>100% Khronos defined Round-trip lossless conversion</td>
</tr>
<tr>
<td>Compute Constructs</td>
<td>Metadata/Intrinsics</td>
<td>Metadata/Intrinsics</td>
<td>Native</td>
</tr>
<tr>
<td>Graphics Constructs</td>
<td>No</td>
<td>No</td>
<td>Native</td>
</tr>
<tr>
<td>Supported Language Feature Set</td>
<td>OpenCL C 1.2</td>
<td>OpenCL C 1.2 / 2.0</td>
<td>OpenCL 2.1 Core</td>
</tr>
<tr>
<td>OpenCL Ingestion</td>
<td>OpenCL 1.2 Extension</td>
<td>OpenCL 2.0 Extension</td>
<td>OpenCL 2.1 Core</td>
</tr>
<tr>
<td>Vulkan Ingestion</td>
<td>-</td>
<td>-</td>
<td>Vulkan Core</td>
</tr>
</tbody>
</table>
SPIR-V at the Center of Language Ecosystem

* Khronos considering developing open source implementations of these translators

- **OpenCL C**
  - **OpenCL C++**
  - **GLSL**
  - **New kernel and shader Languages**

- **Other Intermediate Forms**
  - **LLVM**

- **Other Languages**
  - **OpenCL Driver A**
  - **OpenCL Driver B**
  - **Vulkan Driver X**
  - **Vulkan Driver Y**

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

* Khronos considering developing open source implementations of these translators

- **OpenCL C++**
- **GLSL**
- **New kernel and shader Languages**
- **Other Languages**
- **OpenCL Driver A**
- **OpenCL Driver B**
- **Vulkan Driver X**
- **Vulkan Driver Y**

© Copyright Khronos Group 2015 - Page 16
Ground-up Explicit API Redesign

<table>
<thead>
<tr>
<th>OpenGL.</th>
<th>Vulkan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Originally architected for graphics workstations with direct renderers and split memory</td>
<td>Matches architecture of modern platforms including mobile platforms with unified memory, tiled rendering</td>
</tr>
<tr>
<td>Driver does lots of work: state validation, dependency tracking, error checking. Limits and randomizes performance</td>
<td>Explicit API – the application has direct, predictable control over the operation of the GPU</td>
</tr>
<tr>
<td>Threading model doesn’t enable generation of graphics commands in parallel to command execution</td>
<td>Multi-core friendly with multiple command buffers that can be created in parallel</td>
</tr>
<tr>
<td>Syntax evolved over twenty years – complex API choices can obscure optimal performance path</td>
<td>Removing legacy requirements simplifies API design, reduces specification size and enables clear usage guidance</td>
</tr>
<tr>
<td>Shader language compiler built into driver. Only GLSL supported. Have to ship shader source</td>
<td>SPIR-V as compiler target simplifies driver and enables front-end language flexibility and reliability</td>
</tr>
<tr>
<td>Despite conformance testing developers must often handle implementation variability between vendors</td>
<td>Simpler API, common language front-ends, more rigorous testing increase cross vendor functional/performance portability</td>
</tr>
</tbody>
</table>
SPIR-V Advantages for Developers

- Developers can use same front-end compiler across multiple platforms
  - Eliminating major source of cross-vendor portability
- Reduces runtime shader/kernel compilation time
  - Driver only has to process SPIR-V not full source language
- Don’t have to ship shader/kernel source code
  - Provides a measure of IP protection
- Drivers are simpler and more reliable
  - No need to include front-end compilers
- SPIR-V Whitepaper
Vulkan Status

- Rapid progress since project start in June 2014
  - Significant proposals and IP contributions received from members

- Participants come from all segments of the graphics industry
  - Including an unprecedented level of participation from game engine ISVs

- Initial specs and implementations expected this year
  - Will work on any GPU hardware that supports OpenGL ES 3.1/OpenGL 4.X and up
  - Can ship on any OS - including Windows XP/7/8
Virtual Reality Will Influence Graphics APIs

- The ability to generate ‘Presence’ is becoming achievable at reasonable cost
  - Using visual input to generate subconscious belief in a virtual situation

- PC-based AND mobile systems
  - Beginning to enable developer experimentation

- VR Requirements will affect how graphics APIs generate visual imagery
  - Control over generation of stereo pairs - slightly different view for each eye
  - Optical system geometric correction in rendering path
  - Reduce latency through elimination of in-driver buffering
  - Asynchronously warp framebuffer for instantaneous response to head movement

Many companies driving the VR revolution are participating in the Vulkan working group
Khronos Open Standards for Graphics and Compute
A comprehensive family of APIs to address the full spectrum of developer requirements

1990’s
Workhorse cross-platform API for professional 3D apps and gaming

2000’s
Ubiquitous API for mobile gaming and general purpose graphics

2008
Heterogeneous parallel computation

Portable intermediate representation for graphics and parallel compute

2014
High-efficiency GPU graphics and compute API for performance critical apps

2015
Vulkan™

All APIs will be evolved and maintained to meet industry needs
Rich mix of open technologies for future innovation