Android

Vulkan on Android

2024 Vulkanised

7 February, 2024
Ian Elliott
Software Engineer, Google
ianelliott@google.com
Topics

1. Android Vulkan Strategy
2. Getting Consistent Vulkan Features
3. Getting Correctness & Stability
4. Eliminating Performance Surprises
Stay tuned for Google I/O

More details will be discussed there
Topic 1

Android Vulkan Strategy
Android Vulkan Strategy

Everything works well on top of Vulkan!
Not All Software Will Use Vulkan Directly

Most will continue using Engines, Middleware, and Layered APIs

For example: Unity, Unreal, Godot, ANGLE, WebGPU, HWUI, and SkiaVk
Not All Software Will Use Vulkan Directly

Most will continue using Engines, Middleware, and Layered APIs

For example: Unity, Unreal, Godot, ANGLE, WebGPU, HWUI, and SkiaVk

- For developers of engines, middleware, and layered APIs:
  - Create a Vulkan backend
  - Make Vulkan backend the default
    - Let us know when/why if it can’t be used
  - Eventually delete OpenGL ES (GLES) backend
    - We’ll talk about dependencies for this
ANGLE for the Stragglers

ANGLE is a layered implementation of OpenGL ES (GLES)

- Vulkan Backend is GLES 3.2 conformant on some phones today
- ANGLE is the GLES driver for newer Exynos versions of Galaxy phones
  - Expect more and more phones using ANGLE
- Can reduce fragmentation and effort across the ecosystem
- Good long-term solution for older applications that:
  - Directly use GLES
  - Won’t be updated
What Developers Need to Work Well on Vulkan

Consistent Features
Correctness
Stability
No Performance Surprises
Topic 2

Getting Consistent Vulkan Features
Getting Consistent Features on Android

Android is using Vulkan Profiles

● What are Vulkan Profiles?
  • JSON file that identifies required functionality:
    ■ Core Version (e.g. Vulkan 1.3)
    ■ Additional required feature bits
    ■ Additional required Extensions
    ■ Additional required Fonts
    ■ Higher driver limits
● Software developers can create their own profiles
Two Types of Android Vulkan Profiles

Android Baseline Profile (ABP)

- First attempt at Android profiles:
  - ABP2021
  - ABP2022
- Backwards looking:
  - 85+% of active devices at time it was created
  - Reflects random choices of HW developers
- Expect no new ABPs
## Two Types of Android Vulkan Profiles

<table>
<thead>
<tr>
<th>Android Baseline Profile (ABP)</th>
<th>Vulkan Profiles for Android (VPA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>- First attempt at Android profiles:</td>
<td>- New with Android 15</td>
</tr>
<tr>
<td>- ABP2021</td>
<td>- VPA15_minimums</td>
</tr>
<tr>
<td>- ABP2022</td>
<td>- Forwards looking:</td>
</tr>
<tr>
<td>- Backwards looking:</td>
<td>- Reflects needs of SW developers</td>
</tr>
<tr>
<td>- 85+% of active devices at time it was created</td>
<td>- Driving consistent features as soon as HW developers can deliver them</td>
</tr>
<tr>
<td>- Reflects random choices of HW developers</td>
<td></td>
</tr>
<tr>
<td>- Expect no new ABPs</td>
<td></td>
</tr>
</tbody>
</table>
## Two Types of Android Vulkan Profiles

<table>
<thead>
<tr>
<th>Android Baseline Profile (ABP)</th>
<th>Vulkan Profiles for Android (VPA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>- First attempt at Android profiles:</td>
<td>- New with Android 15</td>
</tr>
<tr>
<td>- ABP2021</td>
<td>- <strong>VPA15_minimums</strong></td>
</tr>
<tr>
<td>- ABP2022</td>
<td>- Forwards looking:</td>
</tr>
<tr>
<td>- Backwards looking:</td>
<td>- Reflects needs of SW developers</td>
</tr>
<tr>
<td>- 85+% of active devices at time it was created</td>
<td>- Driving consistent features as soon as HW developers can deliver them</td>
</tr>
<tr>
<td>- Reflects random choices of HW developers</td>
<td>- <strong>Required of all newly-launching SoCs for a given Android release</strong></td>
</tr>
<tr>
<td>- Expect no new ABPs</td>
<td></td>
</tr>
</tbody>
</table>
Two Types of Android Vulkan Profiles

Android Baseline Profile (ABP)
- First attempt at Android profiles:
  - ABP2021
  - ABP2022
- Backwards looking:
  - 85+% of active devices at time it was created
  - Reflects random choices of HW developers
- Expect no new ABPs

Vulkan Profiles for Android (VPA)
- New with Android 15
  - VPA15_minimums
- Forwards looking:
  - Reflects needs of SW developers
  - Driving consistent features as soon as HW developers can deliver them
- Required of all newly-launching SoCs for a given Android release
- Expect new VPAs every year
Input. Software developers tell us what features they need and why.

Coalesce. Android coalesces all the input and proposes profile contents to hardware developers.

Feedback. GPU developers tell us if/when features can be delivered for which GPUs. We work with SoC developers to encourage GPUs/drivers that deliver profile contents.

Iterative Refinement. We work several releases in advance because of long hardware lead times.

Publish. The VPA for the next Android release is put on GitHub.
If you are an Android software developer, please provide us with input.

Please Get Involved!
Topic 3

Getting Correctness & Stability
Getting Correctness & Stability

Need the Following:

- Improving drivers, applications, engines, etc.
- Getting updated drivers
Vicious Cycle

IHV Tests with Known Content

IHV Ships Driver to OEM

OEM ships new system driver
Vicious Cycle

Software Developer:
- Suspect a driver bug
  - Who to report it to?
  - I don’t have time to deal with this!
  - I need to make money by shipping my game

OEM ships new system driver

ISV Suspects a Driver Bug

IHV Tests with Known Content

Oh no! My game isn’t working on <device>
Vicious Cycle

Software Developer:
- Suspect a driver bug
  - Who to report it to?
  - I don’t have time to deal with this!
  - I need to make money by shipping my game
- Ship with GLES (again)

IHV Ships Driver to OEM

OEM ships new system driver

IHV Tests with Known Content

ISV Suspects a Driver Bug

ISV Ships with GLES
Vicious Cycle

Software Developer:
- Suspect a driver bug
  - Who to report it to?
  - I don’t have time!
- Ship with GLES (again)

Driver Developer:
- Driver quality must be good because:
  - We run lots of tests
    - No bugs seen
  - No bugs reported

OEM ships new system driver

ISV Suspects a Driver Bug

ISV Ships with GLES

IHV Tests with Known Content

IHV Ships Driver to OEM

No Bug Reported to IHV
Vicious Cycle

Software Developer:
- Suspect a driver bug
  - Who to report it to?
  - I don’t have time!
- Ship with GLES (again)

Driver Developer:
- Driver quality must be good because:
  - We run lots of tests
    - No bugs seen
  - No bugs reported

It Gets Worse:
- Many OEMs Don’t Ship Updated Drivers

OEM Ships OLD driver

ISV Ships with GLES

ISV Suspects a Driver Bug

IHV Tests with Known Content

No Bug Reported to IHV

Android
Drivers Improve with Exposure to Apps
Apps Improve with Exposure to Drivers
Drivers Improve with Exposure to Apps
Apps Improve with Exposure to Drivers
Breaking the Vicious Cycle

With a new Virtuous Cycle:

- Make it easy for developers to report suspected driver bugs

- OEM ships new system driver
- ISV Captures Trace of Bug
- ISV Reports Driver Bug
- Google Works with IHVs
With a new Virtuous Cycle:

- Make it easy for developers to report suspected driver bugs
- Pre-release drivers for developers to ensure bugs are fixed
- Mechanism for developers to opt-in to use an optional driver with bug fixes
With a new Virtuous Cycle:

- Make it easy for developers to report suspected driver bugs
- Pre-release drivers for developers to ensure bugs are fixed
- Mechanism for developers to opt-in to use an optional driver with bug fixes
Application-Based Tests

Need New Tests that Reflect what Engines/Applications do

- Vulkan CTS isn’t sufficient
  - More of a collection of unit tests
  - Need tests based on application usage patterns
Application-Based Tests

Need New Tests that Reflect what Engines/Applications do

- Vulkan CTS isn’t sufficient
  - More of a collection of unit tests
  - Need tests based on application usage patterns

- Inspiration:
  - ANGLE can be used to create portable traces of GLES content
Application-Based Tests

Need New Tests that Reflect what Engines/Applications do

- Vulkan CTS isn’t sufficient
  - More of a collection of unit tests
  - Need tests based on application usage patterns
- Inspiration:
  - ANGLE can be used to create portable traces of GLES content
- GFXReconstruct traces are inherently non-portable
  - Look at parts of traces to see unique ways Vulkan is used
  - From these, derive tests that are better than Vulkan CTS is today
Application-Based Tests

Need New Tests that Reflect what Engines/Applications do

- Vulkan CTS isn’t sufficient
  - More of a collection of unit tests
  - Need tests based on application usage patterns
- Inspiration:
  - ANGLE can be used to create portable traces of GLES content
- GFXReconstruct traces are inherently non-portable
  - Look at parts of traces to see unique ways Vulkan is used
  - From these, derive tests that are better than Vulkan CTS is today
- Please provide us with traces, code snippets, etc. that we can turn into tests
Topic 4

Eliminating Performance Surprises
Eliminating Performance Surprises

Shift Performance Focus from GLES to Vulkan

- We are working with partners on newer versions of our profiling tools
  - Give you visibility into where the time is going
  - Provide actionable insights
Eliminating Performance Surprises

Shift Performance Focus from GLES to Vulkan

- We are working with partners on newer versions of our profiling tools
  - Give you visibility into where the time is going
  - Provide actionable insights
- Replace old GLES-based benchmarks with Vulkan-based benchmarks
  - Industry-standard benchmarks
  - Create benchmarks that measure what your application/engine does
    - Please create and share your own, or
    - Provide us with traces, code snippets, etc. that we can use
Thank you