AMD GPU Performance Revealed

Radeon™ GPU Profiler
Microsoft® PIX for Windows
Radeon™ GPU Analyzer

Rodrigo Urra, Amit Ben-Moshe

Advanced Micro Devices, Inc.
AGENDA

Radeon™ GPU Profiler

Rodrigo Urra

Introduction

Tour of last year

What’s new and coming
What is Radeon GPU Profiler?

**GPU performance analysis tool**
- Shows low level profiling data
- Visualizes GPU workloads
- Designed to identify performance bottlenecks
- Helps bridge the gap between explicit APIs and GCN

**Designed to support**
- Linux® and Windows®
- Vulkan® and DirectX® 12
GDC 2018 - RGP 1.2

Overview
- Frame summary
- Most expensive events
- Barriers
- Context rolls

Events
- Wavefront occupancy
- Event timing
- Pipeline state

RenderDoc interop
RGP 1.2 - Frame summary

Visualize command buffer submission, queue parallelism, and synchronization
RGP 1.2 - Frame summary

Visualize command buffer submission, queue parallelism, and synchronization
RGP 1.2 - Frame summary

Visualize command buffer submission, queue parallelism, and synchronization
RGP 1.2 - Frame summary

Visualize command buffer submission, queue parallelism, and synchronization
RGP 1.2 - Most expensive events

Pinpoint optimization candidates
RGP 1.2 - Barriers

Spot potential pipeline bubbles
RGP 1.2 - Context rolls

There were 446 context rolls from 1368 draw calls. That's an average of 2.93 draw calls per context. None of these context rolls were completely redundant.

State which causes context rolls

<table>
<thead>
<tr>
<th>State</th>
<th>Context rolls</th>
<th>Redundant</th>
<th>Completely redundant</th>
<th>Reclusive</th>
</tr>
</thead>
<tbody>
<tr>
<td>renderbufferimagestable</td>
<td>234</td>
<td>47</td>
<td>0 (VS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>polygondirty</td>
<td>47</td>
<td>47 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>cutmode</td>
<td>47</td>
<td>8 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>frontface</td>
<td>47</td>
<td>17 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>depthEnabled</td>
<td>30</td>
<td>6 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>depthEnabledLamp</td>
<td>10</td>
<td>16 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>depthEnabledInspector</td>
<td>30</td>
<td>8 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>lineWidth</td>
<td>11</td>
<td>11 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>VAPOR/Geometry/IsSamplestate</td>
<td>451</td>
<td>413 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>VAPOR/Geometry/IsSamplestate</td>
<td>451</td>
<td>413 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>VAPOR/Geometry/IsSamplestate</td>
<td>54</td>
<td>14 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
<tr>
<td>VAPOR/Geometry/IsSamplestate</td>
<td>54</td>
<td>14 (RS)</td>
<td>0 (RS)</td>
<td>0 (RS)</td>
</tr>
</tbody>
</table>

Events

<table>
<thead>
<tr>
<th>Queue index</th>
<th>Event ID</th>
<th>Event</th>
<th>Duration</th>
<th>Stall due to context rolls</th>
</tr>
</thead>
<tbody>
<tr>
<td>1218</td>
<td>AcvDrawIndex(208, 1, 0, 0)</td>
<td>543.104 ms</td>
<td>100.187 ms</td>
<td></td>
</tr>
<tr>
<td>1220</td>
<td>AcvDrawIndex(208, 1, 0, 0)</td>
<td>543.743 ms</td>
<td>66.688 ms</td>
<td></td>
</tr>
<tr>
<td>1222</td>
<td>AcvDrawIndex(208, 1, 0, 0)</td>
<td>178.981 ms</td>
<td>65.905 ms</td>
<td></td>
</tr>
<tr>
<td>1225</td>
<td>AcvDrawIndexed(208, 1, 0, 0)</td>
<td>888.080 ms</td>
<td>24.171 ms</td>
<td></td>
</tr>
<tr>
<td>1232</td>
<td>AcvDrawIndex(208, 1, 0, 0)</td>
<td>264.603 ms</td>
<td>4.018 ms</td>
<td></td>
</tr>
<tr>
<td>1476</td>
<td>CnvDenseFastColorClear()</td>
<td>143.713 ms</td>
<td>0 ms</td>
<td></td>
</tr>
<tr>
<td>1422</td>
<td>AcvDrawIndexed(315, 1, 0, 0)</td>
<td>57.247 ms</td>
<td>0 ms</td>
<td></td>
</tr>
<tr>
<td>1419</td>
<td>AcvDrawL(1, 0, 0)</td>
<td>283.956 ms</td>
<td>0 ms</td>
<td></td>
</tr>
<tr>
<td>1416</td>
<td>AcvDrawL(1, 0, 0)</td>
<td>749.502 ms</td>
<td>0 ms</td>
<td></td>
</tr>
<tr>
<td>1413</td>
<td>AcvDrawL(1, 0, 0)</td>
<td>874.795 ms</td>
<td>0 ms</td>
<td></td>
</tr>
</tbody>
</table>

Find redundant state changes
RGP 1.2 - Wavefront occupancy

See how busy the GPU is
RGP 1.2 - Wavefront occupancy

See how busy the GPU is
RGP 1.2 - Wavefront occupancy

See how busy the GPU is
See how busy the GPU is
RGP 1.2 - Wavefront occupancy

See how busy the GPU is
RGP 1.2 - Event timing

Understand the cost of GPU work in your frame
RGP 1.2 - Pipeline state

Exposé pipeline state and GPR consumption
RGP 1.2 - RenderDoc ↔ RGP interop

Understand correlation between rendered scene ↔ GPU work
Cool, but I want more!

What about

– Wavefront shader execution?
– How do my shaders translate to GCN ISA?
– Can you give me information about my render targets?
– Enhanced user marker visualization?
– What if I’m a compute-only developer?
So what’s new?

GDC 2018

RGP 1.2
RenderDoc interop
So what’s new?

GDC 2018

RGP 1.2
RenderDoc interop

RGP 1.3
Render & depth targets

RGP 1.4
OpenCL™ support
• Instruction trace
• Shader ISA
• User marker display

RGP 1.5
Render & depth targets
So what’s new?

GDC 2018

RGP 1.2
   RenderDoc interop

RGP 1.3
   Render & depth targets

RGP 1.4
   OpenCL™ support

OpenCL™ support
   Instruction trace
   Shader ISA
   User marker display

RGP 1.5

GDC 2019
So what’s new?

GDC 2018

- RGP 1.2
  - RenderDoc interop

GDC 2019

- RGP 1.3
  - Render & depth targets
- RGP 1.4
  - OpenCL™ support
- RGP 1.5.x
  - Instruction timing
  - Shader ISA
  - User marker display
RGP 1.3 - Render & depth targets

Understand all your render targets
Interactive timeline

- Three sections - color, depth, other
- Displays when gfx/compute passes begin/end
- Hooked up to other RGP panes via context menu
RGP 1.3 - Render & depth targets

Interactive timeline
- Three sections - color, depth, other
- Displays when gfx/compute passes begin/end
- Hooked up to other RGP panes via context menu
Interactive timeline
- Three sections - color, depth, other
- Displays when gfx/compute passes begin/end
- Hooked up to other RGP panes via context menu
Interactive timeline
- Three sections - color, depth, other
- Displays when gfx/compute passes begin/end
- Hooked up to other RGP panes via context menu
## RGP 1.3 - Render & depth targets

<table>
<thead>
<tr>
<th>Name</th>
<th>Format</th>
<th>Width</th>
<th>Height</th>
<th>Draw calls</th>
<th>Compression</th>
<th>Sample count</th>
<th>Out of order draw calls</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color RT #0</td>
<td>VK_FORMAT_R16G16B16A16_FLOAT</td>
<td>1920</td>
<td>1080</td>
<td>33</td>
<td>ON</td>
<td>1</td>
<td>0 / 33</td>
<td>680.296 µs</td>
</tr>
<tr>
<td>Color RT #1</td>
<td>VK_FORMAT_R16G16B16A16_FLOAT</td>
<td>1920</td>
<td>1080</td>
<td>30</td>
<td>ON</td>
<td>1</td>
<td>0 / 30</td>
<td>365.772 µs</td>
</tr>
<tr>
<td>Color RT #2</td>
<td>VK_FORMAT_R16G16_Float</td>
<td>1920</td>
<td>1080</td>
<td>30</td>
<td>ON</td>
<td>1</td>
<td>0 / 30</td>
<td>365.772 µs</td>
</tr>
<tr>
<td>Depth RT #0</td>
<td>VK_FORMAT_D32_FLOAT</td>
<td>1920</td>
<td>1080</td>
<td>52</td>
<td>ON</td>
<td>1</td>
<td>0 / 52</td>
<td>3,667.100 µs</td>
</tr>
<tr>
<td>Depth RT #1</td>
<td>VK_FORMAT_D32_FLOAT</td>
<td>512</td>
<td>512</td>
<td>406</td>
<td>ON</td>
<td>1</td>
<td>406 / 406 (100.0%)</td>
<td>711.623 µs</td>
</tr>
<tr>
<td>Color RT #3</td>
<td>VK_FORMAT_R16G16B16A16_FLOAT</td>
<td>1920</td>
<td>1080</td>
<td>13</td>
<td>ON</td>
<td>1</td>
<td>0 / 13</td>
<td>2,140.486 µs</td>
</tr>
<tr>
<td>Color RT #4</td>
<td>VK_FORMAT_R16G16B16A16_FLOAT</td>
<td>960</td>
<td>540</td>
<td>2</td>
<td>OFF</td>
<td>1</td>
<td>0 / 2</td>
<td>77.042 µs</td>
</tr>
<tr>
<td>Color RT #5</td>
<td>VK_FORMAT_R16G16B16A16_FLOAT</td>
<td>480</td>
<td>270</td>
<td>1</td>
<td>OFF</td>
<td>1</td>
<td>0 / 1</td>
<td>17.560 µs</td>
</tr>
<tr>
<td>Color RT #6</td>
<td>VK_FORMAT_R16G16B16A16_FLOAT</td>
<td>480</td>
<td>270</td>
<td>1</td>
<td>ON</td>
<td>1</td>
<td>0 / 1</td>
<td>9.043 µs</td>
</tr>
<tr>
<td>Color RT #7</td>
<td>VK_FORMAT_R16G16B16A16_FLOAT</td>
<td>480</td>
<td>270</td>
<td>1</td>
<td>ON</td>
<td>1</td>
<td>0 / 1</td>
<td>7.634 µs</td>
</tr>
<tr>
<td>Color RT #8</td>
<td>VK_FORMAT_R16G16B16A16_FLOAT</td>
<td>1920</td>
<td>1080</td>
<td>1</td>
<td>OFF</td>
<td>1</td>
<td>0 / 1</td>
<td>65.890 µs</td>
</tr>
<tr>
<td>Color RT #9</td>
<td>VK_FORMAT_R16G16B16A16_FLOAT</td>
<td>1920</td>
<td>1080</td>
<td>1</td>
<td>OFF</td>
<td>1</td>
<td>0 / 1</td>
<td>164.295 µs</td>
</tr>
</tbody>
</table>

### Properties table

- Dimensions and format
- Draw and sample counts
- Duration
- Optimizations
What is it?
- RGP minus the graphics-only stuff

How do I use it?
- Developer panel now detects OpenCL™ applications
- Set number of kernels, and capture!
Planned for RGP 1.5.x

### Instruction timing

**Wavefront occupancy**

- **Event timing**

- **Pipeline state**

**Instruction timing**

*Note: The image contains a graph and code snippets for instruction timing, wavefront occupancy, and pipeline states.*

### Shader ISA

- **User marker display**

*Note: The image contains a screenshot of a shader ISA and user marker display.*
Planned for RGP 1.5.x - Instruction timing

- Top-down program execution
- Find which part of your program is hot
- Available for all shader stages
- See instruction durations
- Functional unit utilization (VALU, SALU, LDS)
- Does not require shader modification
Planned for RGP 1.5.x - Instruction timing

- Top-down program execution
- Find which part of your program is hot
- Available for all shader stages
- See instruction durations
- Functional unit utilization (VALU, SALU, LDS)
- Does not require shader modification
Planned for RGP 1.5.x - Instruction timing

- Top-down program execution
- Find which part of your program is hot
- Available for all shader stages
- See instruction durations
- Functional unit utilization (VALU, SALU, LDS)
- Does not require shader modification
Planned for RGP 1.5.x - Instruction timing

- Top-down program execution
- Find which part of your program is hot
- Available for all shader stages
- See instruction durations
- Functional unit utilization (VALU, SALU, LDS)
- Does not require shader modification
Planned for RGP 1.5.x - Shader ISA

See shader code in pipeline state
Planned for RGP 1.5.x - User marker display

Know what the GPU is working on
Planned for RGP 1.5.x - User marker display

Know what the GPU is working on
AGENDA

Microsoft® PIX for Windows
Rodrigo Urra

Introduction
AMD-specific GPU data
What is Microsoft® PIX for Windows?

Graphics debugging and perf tuning tool
– DirectX® 12 on Windows 10
What is Microsoft® PIX for Windows?

Graphics debugging and perf tuning tool
– DirectX® 12 on Windows 10
What is Microsoft® PIX for Windows?

Graphics debugging and perf tuning tool
– DirectX® 12 on Windows 10
Microsoft® PIX for Windows – AMD support

Graphics debugging and perf tuning tool
- DirectX® 12 on Windows 10

AMD-specific GPU data
- Event list counters
### Event list counters: Command-specific counters

<table>
<thead>
<tr>
<th>Category</th>
<th>CBColorAndMaskWritten</th>
<th>CBMemWritten</th>
<th>FetchSize</th>
<th>L1CacheHitCount</th>
<th>L1CacheMissCount</th>
<th>L2Cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorBuffer</td>
<td>0</td>
<td>0</td>
<td>734,720</td>
<td>101,208</td>
<td>14,176</td>
<td></td>
</tr>
<tr>
<td>CommandRead</td>
<td>0</td>
<td>0</td>
<td>27,744</td>
<td>0</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>CommandWrite</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>827,968</td>
<td>26,497</td>
<td>13,969</td>
</tr>
<tr>
<td>ComputeShaderCompressed</td>
<td>0</td>
<td>0</td>
<td>103,360</td>
<td>2,150</td>
<td>1,092</td>
<td></td>
</tr>
<tr>
<td>ComputeShaderUncompressed</td>
<td>0</td>
<td>0</td>
<td>168,064</td>
<td>9,893</td>
<td>2,035</td>
<td></td>
</tr>
<tr>
<td>DepthStencilBuffer</td>
<td>0</td>
<td>0</td>
<td>629,696</td>
<td>9,194</td>
<td>7,101</td>
<td></td>
</tr>
<tr>
<td>GlobalMemory</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>83,048</td>
<td>4,343,124</td>
<td>340,342</td>
</tr>
<tr>
<td>HullShader</td>
<td>0</td>
<td>0</td>
<td>2,651,648</td>
<td>33,951,739</td>
<td>919,645</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>8,411,488</td>
<td>24,135,648</td>
<td>819,645</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>8,411,488</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>899,120</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>30,624</td>
<td>30,624</td>
<td>317,280</td>
<td>104,347</td>
<td>7,215</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>83,048</td>
<td>83,048</td>
<td>1,199,024</td>
<td>283,204</td>
<td>24,084</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>542,240</td>
<td>542,240</td>
<td>816,648</td>
<td>665,313</td>
<td>25,718</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>616,320</td>
<td>40,287</td>
<td>13,256</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>135,744</td>
<td>135,744</td>
<td>581,216</td>
<td>329,573</td>
<td>14,293</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>3,043,040</td>
<td>3,043,040</td>
<td>3,431,616</td>
<td>3,790,288</td>
<td>228,053</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>186,240</td>
<td>9,002</td>
<td>3,000</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>575,424</td>
<td>575,424</td>
<td>998,848</td>
<td>366,123</td>
<td>27,752</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>938,016</td>
<td>938,016</td>
<td>1,813,568</td>
<td>1,180,494</td>
<td>79,544</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>190,784</td>
<td>190,784</td>
<td>682,599</td>
<td>85,208</td>
<td>11,840</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>1,728</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>93,024</td>
<td>93,024</td>
<td>999,168</td>
<td>172,500</td>
<td>17,260</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>63,488</td>
<td>3,213</td>
<td>1,025</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>2,069,024</td>
<td>2,069,024</td>
<td>2,538,976</td>
<td>4,355,959</td>
<td>223,388</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>1,137,536</td>
<td>56,958</td>
<td>17,840</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>281,792</td>
<td>14,303</td>
<td>4,565</td>
<td></td>
</tr>
<tr>
<td>MemoryGroup</td>
<td>0</td>
<td>0</td>
<td>2,816</td>
<td>103</td>
<td>32</td>
<td></td>
</tr>
</tbody>
</table>
Microsoft® PIX for Windows – AMD support

Graphics debugging and perf tuning tool
  – DirectX® 12 on Windows 10

AMD-specific GPU data
  – Event list counters
  – Wavefront occupancy (SQTT)
Wavefront occupancy - Execution stage utilization
Microsoft® PIX for Windows – AMD support

Graphics debugging and perf tuning tool
- DirectX® 12 on Windows 10

AMD-specific GPU data
- Event list counters
- Wavefront occupancy (SQTT)
- High frequency counters (SPM)
High frequency counters (HFC): ns execution sampling
Microsoft® PIX for Windows – AMD support

Graphics debugging and perf tuning tool
- DirectX® 12 on Windows 10

AMD-specific GPU data
- Event list counters
- Wavefront occupancy (SQTT)
- High frequency counters (SPM)

Displayed data
- Includes driver + GPU work

Supported GPUs
Radeon™ VII, Radeon™ Vega, Radeon™ RX400 and RX500, and Intel Core i7
Radeon™ RX Vega M
AGENDA

Radeon™ GPU Analyzer

Amit Ben-Moshe

Introduction

RGA for Vulkan™

A look at the UI

The RGA layer
What is Radeon GPU Analyzer

Compiler and code analysis tool

- Generate GCN ISA and IL disassembly
- Retrieve shader HW resource usage
- Static analysis
- Independent of the installed HW
- Designed to support both Linux® and Windows®
- Designed to support multiple APIs: Vulkan™, DirectX®11, OpenGL™, OpenCL™
Integration with other tools

Integrated into tools you love

- RenderDoc, Shader Playground, CodeXL, Pyramid
New approach for shader optimization

- Uses the live AMD driver compiler rather than offline compiler:
  - What you see is what you get
  - Compiler updates as you update the driver
- Vulkan™ app creates and compiles Vulkan™ pipelines

```
GLSL
  ( glslang )
  SPIR-V
  Shaders + State
```
Pipeline state

- JSON representation
  - VkGraphicsPipelineCreateInfo for graphics (.gpso)
  - VkComputePipelineCreateInfo for compute (.cpso)
- Edit/Compose Vulkan™ state in UI
The new RGA UI

**Startup window**

- Select your API of choice

![Radeon GPU Analyzer](image-url)
Home page

Start by creating a graphics or compute pipeline

Hello, welcome to the Radeon GPU Analyzer!

Start
Create new Vulkan graphics pipeline
Create new Vulkan compute pipeline

Recent
No recent sessions
Open .rga file from explorer...

Help
About RGA
Help manual
Getting started guide
Add shaders

Two choices

- Create template GLSL shader using RGA
- Load or drag & drop existing shader file (GLSL / SPIR-V)
Edit GLSL / SPIR-V code

- Syntax highlighting
- Find text
- Jump to line
- Open header files (GLSL)

```glsl
/* Auto-generated with Radeon GPU Analyzer (RGA).*
#version 450

#extension GL_GOOGLE_include_directive : require
#extension GL_ARB_separate_shader_objects : enable

out gl_PerVertex
{
    vec4 gl_Position;
}

layout(location = 0) out vec3 fragColor;

void main()
{
    float x = -1.0 + float(gl_VertexIndex & 1) << 2;
    float y = -1.0 + float(gl_VertexIndex & 2) << 1;
    gl_Position = vec4(x, y, 0, 1);

    fragColor = vec3(1.0f, 0.0f, 0.0f);
}
```
SPIR-V binary editing

Drop SPIR-V binary in RGA

- Binary disassembled
- Syntax highlighted
- Edit and reassemble
- Revert to original

```
; SPIR-V
; Version: 1.0
; Generator: Khronos Glslang Reference Front End; 1
; Bound: 172
; Schema: 0

OpCapability Shader
OpCapability ImageQuery
%1 = OpExtInstImport "GLSL.std.450"
OpMemoryModel logical GLSL450
OpEntryPoint Fragment %main "main" %inUV %outFragColor
OpExecutionMode %main OriginUpperLeft
OpSource GLSL 450
OpSourceExtension "GL_ARB_separate_shader_objects"
OpSourceExtension "GL_ARB_shading_language_420pack"
OpName %main "main"
OpName %weight "weight"
OpName %tex_offset "tex_offset"
OpName %samplerColor "samplerColor"
OpName %ubo "ubo"
OpMemberName %ubo 0 "blurScale"
OpMemberName %ubo 1 "blurStrength"
OpName %ubo "ubo"
OpName %result "result"
OpName %inUV "inUV"
OpName %i "i"

OpDecorate %samplerColor DescriptorSet 0
OpDecorate %samplerColor Binding 1
OpMemberDecorate %ubo 0 Offset 0
OpMemberDecorate %ubo 1 Offset 4
OpDecorate %ubo Block
OpDecorate %ubo DescriptorSet 0
OpDecorate %ubo Binding 0
OpDecorate %inUV Location 0
```
Configure the pipeline state

- Graphical representation of the Vulkan™ pipeline state as a tree
- Easy search and navigation
- Filter the state
- Edit the state
Configure the build settings

**General**
- Target GPUs:
- Predefined macros:
- Additional include directories:
- Vulkan runtime settings:
- ICD location:
- Enable validation layers
- Front-end compiler settings (glslang):
- Additional glslang options:
- Use alternative front-end compiler package

**Select Target GPUs**

<table>
<thead>
<tr>
<th>Product name</th>
<th>Architecture</th>
<th>Compute capability</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMD Embedded Radeon E9170 Series</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>AMD Embedded Radeon E9171</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>AMD Radeon Pro WX 2100</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>AMD Radeon Pro WX 3100</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>Radeon 500 Series</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>Radeon 540 Series</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>Radeon 550 Series</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>Radeon Pro WX 2100</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>Radeon Pro WX 3100</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>Radeon RX Vega M GH Graphics</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
<tr>
<td>Radeon RX Vega M GL Graphics</td>
<td>Graphics IP6</td>
<td>gbx804</td>
</tr>
</tbody>
</table>
Build the pipeline
Examine the disassembly and resource usage

- Disassembly and resource usage in the same view
- Breaks down each instruction into its components
- Color coding by instruction class
- Resource usage hazards highlighted as warnings
How can I get my app’s shaders into RGA?

**Use the RGA layer – beta feature**

- Implicit Vulkan™ layer
- Serializes state and shaders for all pipelines
- Can intercept specific pipelines if VK_EXT_DEBUG_MARKER is used
Using the RGA Layer

Suggested RGA workflow

- Good idea to use VK_EXT_DEBUG_MARKER to name your objects
- Identify your bottleneck pipeline using a profiler or debugger like RenderDoc
- Suppose pipeline “blurOne” was identified as a bottleneck
- Can you locate your shaders? Just drop them in RGA
- Can’t locate shaders or don’t feel like manually editing the state?
  - Use the RGA layer
Drop serialized files in RGA
Build, tweak GLSL / SPIR-V, tweak pipeline state, repeat!
RGA for Vulkan – key takeaways

- Compile through the live-driver
- Edit GLSL source code or SPIR-V binaries
- Edit Vulkan pipeline state with graphical editor
- Extract state/shaders from apps with the layer – beta feature
- Highlights VMEM, SMEM instructions, and excessive HW resource usage
- Runs on non-AMD configurations as well
Thank you!

Information
GPUOpen: https://gpuopen.com/
RGP: https://gpuopen.com/gaming-product/radeon-gpu-profiler-rgp/
RGA: https://gpuopen.com/gaming-product/radeon-gpu-analyzer-rga/
RenderDoc: https://renderdoc.org
Microsoft® PIX: https://blogs.msdn.microsoft.com/pix

Downloads
RGP: https://github.com/GPUOpen-Tools/RGP/releases
RGA: https://github.com/GPUOpen-Tools/RGA/releases

Acknowledgements
Baldur Karlsson for assistance with RenderDoc-RGP and RGA interop
Gregory Mitrano for RGP content

Questions?

Contact
Rodrigo.Urra@amd.com
Amit.Ben-Moshe@amd.com
Disclaimer and Attribution

The information contained herein is for informational purposes only, and is subject to change without notice. This document may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD’s products are as set forth in a signed agreement between the parties or in AMD’s Standard Terms and Conditions of Sale. GD-18

©2019 Advanced Micro Devices, Inc. All rights reserved, AMD, the AMD Arrow logo, Radeon and combinations thereof are trademarks of Advanced Micro Devices, Inc. Microsoft, Windows and DirectX are registered trademarks of Microsoft Corporation. Linux is a registered trademark of Linus Torvalds. Vulkan is a registered trademark of The Khronos Group, Inc. OpenCL is a trademark of Apple, Inc. used by permission from The Khronos Group. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Bloom.jpg on Slide 67 and Code Excerpted on Slide 68 and 69 are Copyright (c) 2016 Sascha Willems

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.