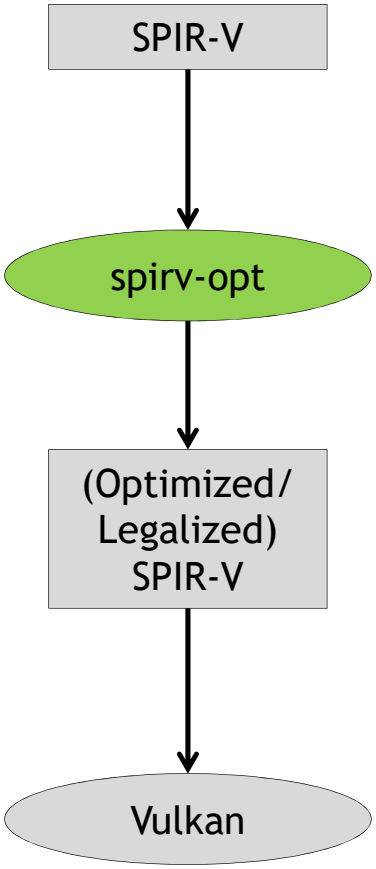
HLSL in Vulkan
There and Back Again

Shrinking and Legalizing Vulkan Shaders with spirv-opt

Greg Fischer, LunarG
January 2017

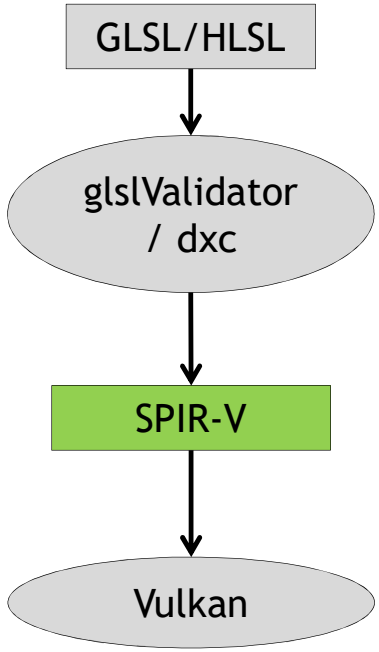
Goals of Talk

- What is spirv-opt?
- What is the status of SPIR-V size?
- What is HLSL Legalization?
- How to engage spirv-opt?



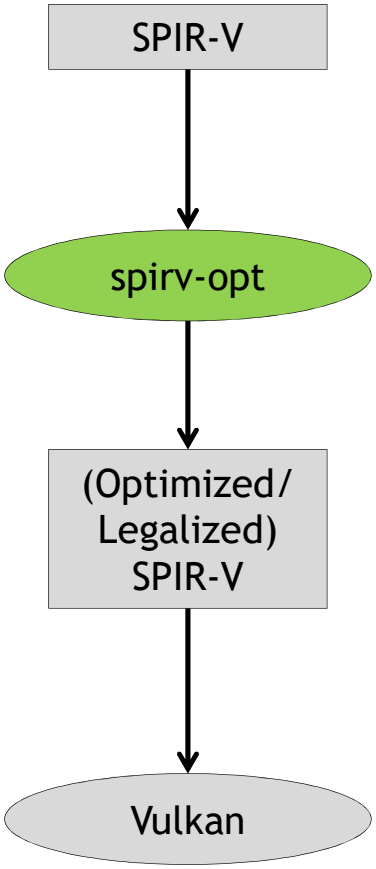
SPIR-V

- Binary shader intermediate representation (IR) for Vulkan (and other APIs)
- SPIR-V is to Vulkan as DX Byte Code is to DirectX
- Primarily generated from high-level shader languages GLSL and HLSL
- Disassembler, Assembler, Validator also available



spirv-opt

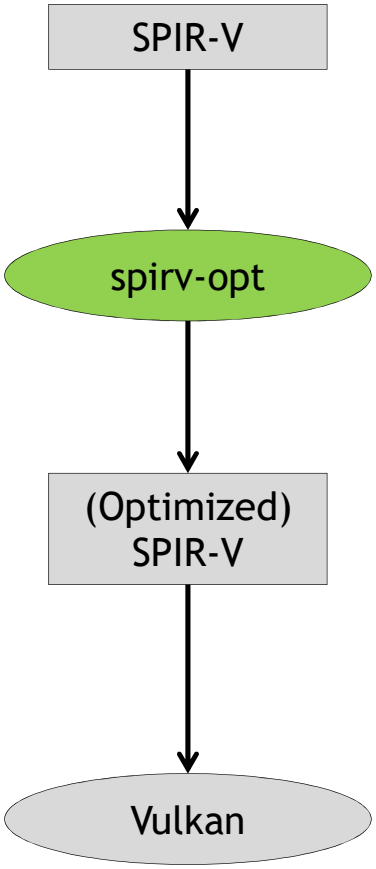
- Open Source (github.com/KhronosGroup/SPIRV-Tools)
- Collaboration between Google and LunarG with support from Valve. Additional contributions from Mesa and Roblox.
- SPIR-V -> “Optimized” SPIR-V
- Goals include
  - Reduced SPIR-V size
  - “Legalized” SPIR-V from HLSL
- Utilizes classic, platform-independent optimization techniques
- First announced at SIGGRAPH, August 2017



spirv-opt: Reducing SPIR-V Size

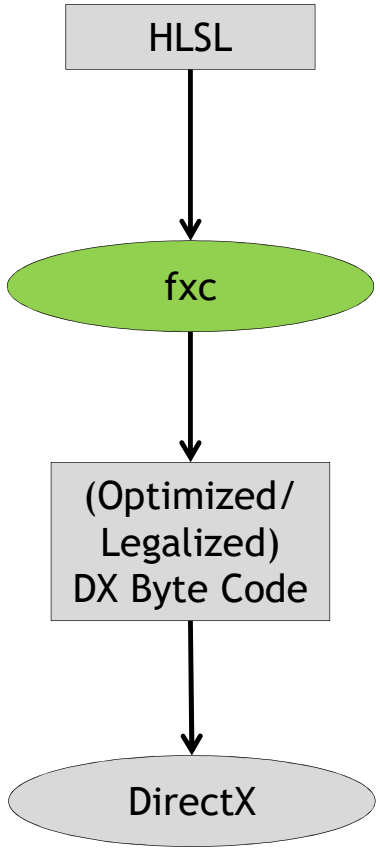
- Optimizations include
  - Function Call Inlining
  - Constant Folding & Propagation
  - Arithmetic Simplification
  - Local Store/Load Elimination
  - Dead Code Elimination
  - Common Subexpression Elimination
  - Debug Information Stripping

- Optimized SPIR-V now within 10% of DX Byte Code size



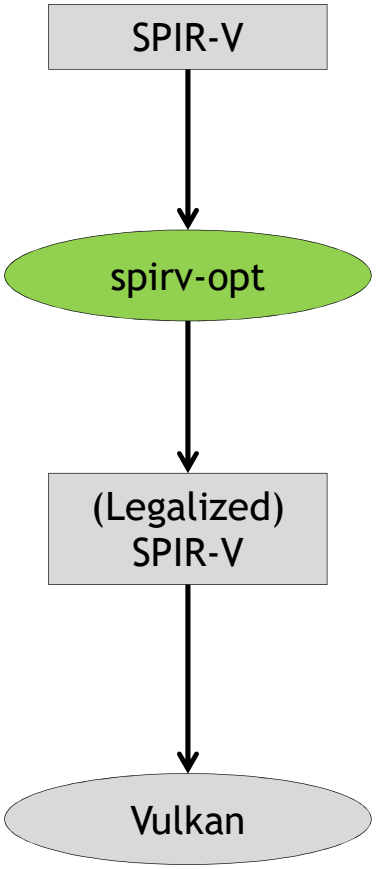
HLSL and fxc Equivalence

- HLSL shaders *must* be “legalized”
- HLSL contains constructs that are not directly supported by graphics hardware
- Optimization techniques can be used to put shader code in a form directly supported by graphics hardware
- fxc is Microsoft’s optimizing shader compiler
- fxc: HLSL -> Highly optimized and "legalized" DX Byte Code
- Current graphics hardware and drivers assume HLSL shaders will have certain optimizations done
- spirv-opt therefore needs to be "equivalent" to fxc in optimization capability when porting DX/HLSL to Vulkan



spirv-opt: HLSL Legalization via Propagation

- Some HLSL features not directly supported in Vulkan (or DX)
  - Opaque structure members (eg. textures, samplers)
  - Local Structured Buffers
- glslangValidator issues warning for problematic constructs: "WARNING: AST will form illegal SPIR-V; need to transform to legalize"
- These constructs are "optimized" away by spirv-opt through function call inlining, dead control flow elimination, value propagation



Opaque Struct Example

```
struct os {
    SamplerState o_s2D;
    Texture2D o_tex;
};

Texture2D tex;
SamplerState s2D;

float4 osCall(os s, float2 f2)
{
    return s.o_tex.Sample(s.o_s2D, f2);
}

float4 main() : SV_TARGET0
{
    os s;
    s.o_tex = tex;
    s.o_s2D = s2D;
    return osCall(s, float2(0.2, 0.3));
}
```

Opaque Struct Example: Optimized

```
Texture2D tex;
SamplerState s2D;



float4 main() : SV_TARGET0
{
    return tex.Sample(s2D, float2(0.2, 0.3));
}
```

Local Structured Buffer Example

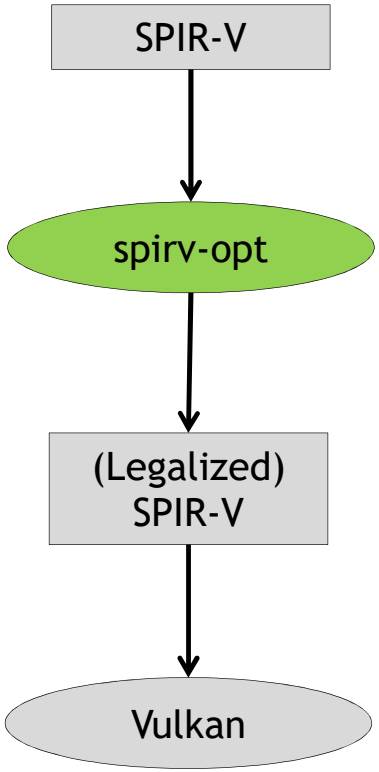
```
struct S {
    float4 f;
};

RWStructuredBuffer<S> gRWSBuffer;

float4 main() : A {
    RWStructuredBuffer<S> t;

    t = gRWSBuffer;

    return t[0].f;
}
```

Local Structured Buffer Example: Optimized

```
struct S {
    float4 f;
};

RWStructuredBuffer<S> gRWSBuffer;

float4 main() : A {
    return gRWSBuffer[0].f;
}
```

spirv-opt: HLSL Legalization via Dead Code Elimination

- HLSL may contain dead texture, sampler, buffer references
- Such dead refs cause Vulkan Validation Layer complaints: "VALIDATION ERROR 0x3d Descriptor set 0x41bde encountered the following validation error at vkCmdDrawIndexed() time: Descriptor in binding #33 at global descriptor index 10 is being used in draw but has not been updated."
- These are optimized away by spirv-opt dead code/branch elimination



Dead Sample Example

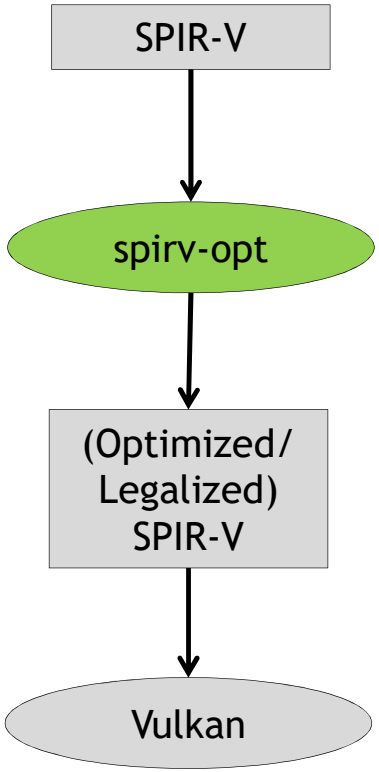
```
Texture2D tex0;
SamplerState s0;

Texture2D tex1;
SamplerState s1;

float4 main() : SV_TARGET0
{
   if (true)
     return tex0.Sample(s0, float2(0.2,0.3));
   else
     return tex1.Sample(s1, float2(0.2,0.3));
}
```

Dead Sample Example: Optimized

```
Texture2D tex0;
SamplerState s0;



float4 main() : SV_TARGET0
{

    return tex0.Sample(s0, float2(0.2,0.3));


}
```

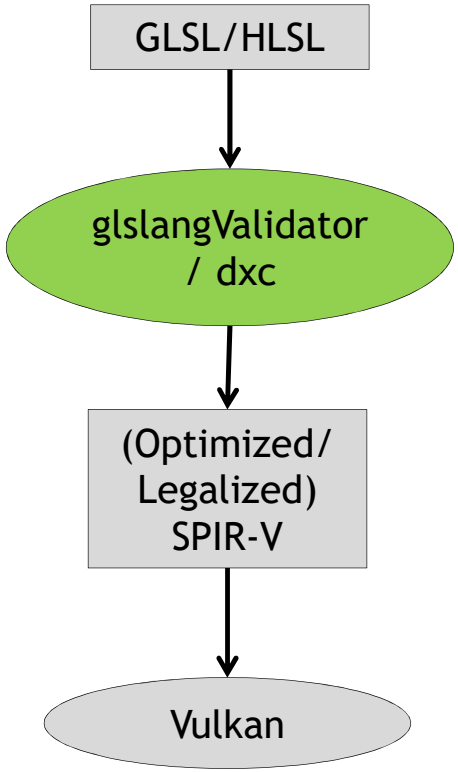
spirv-opt Usage - Direct

- Command line
  - --legalize-hlsl
  - -Os: Optimize for size
  - --<pass> --<pass> ... (see --help)
- API interface
  - Optimizer::RegisterLegalizationPasses()
  - Optimizer::RegisterSizePasses()



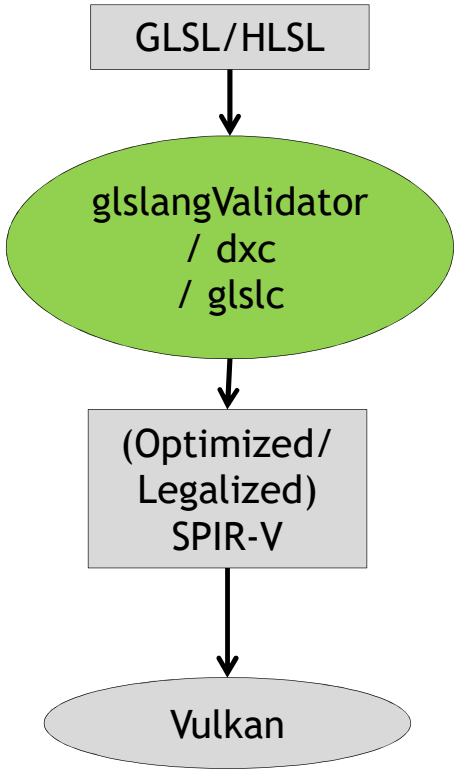
spirv-opt Usage - Indirect through Frontends

- glslangValidator
  - Khronos GLSL/HLSL FE
  - For legacy reasons, glslang does not require SPIRV-Tools to build
  - Legalization by default if built with SPIRV-Tools; can be disabled (see README)
  - -Os: Optimize for size if built with SPIRV-Tools
  - LunarG Vulkan SDK edition built with SPIRV-Tools
  - github.com/KhronosGroup/glslang



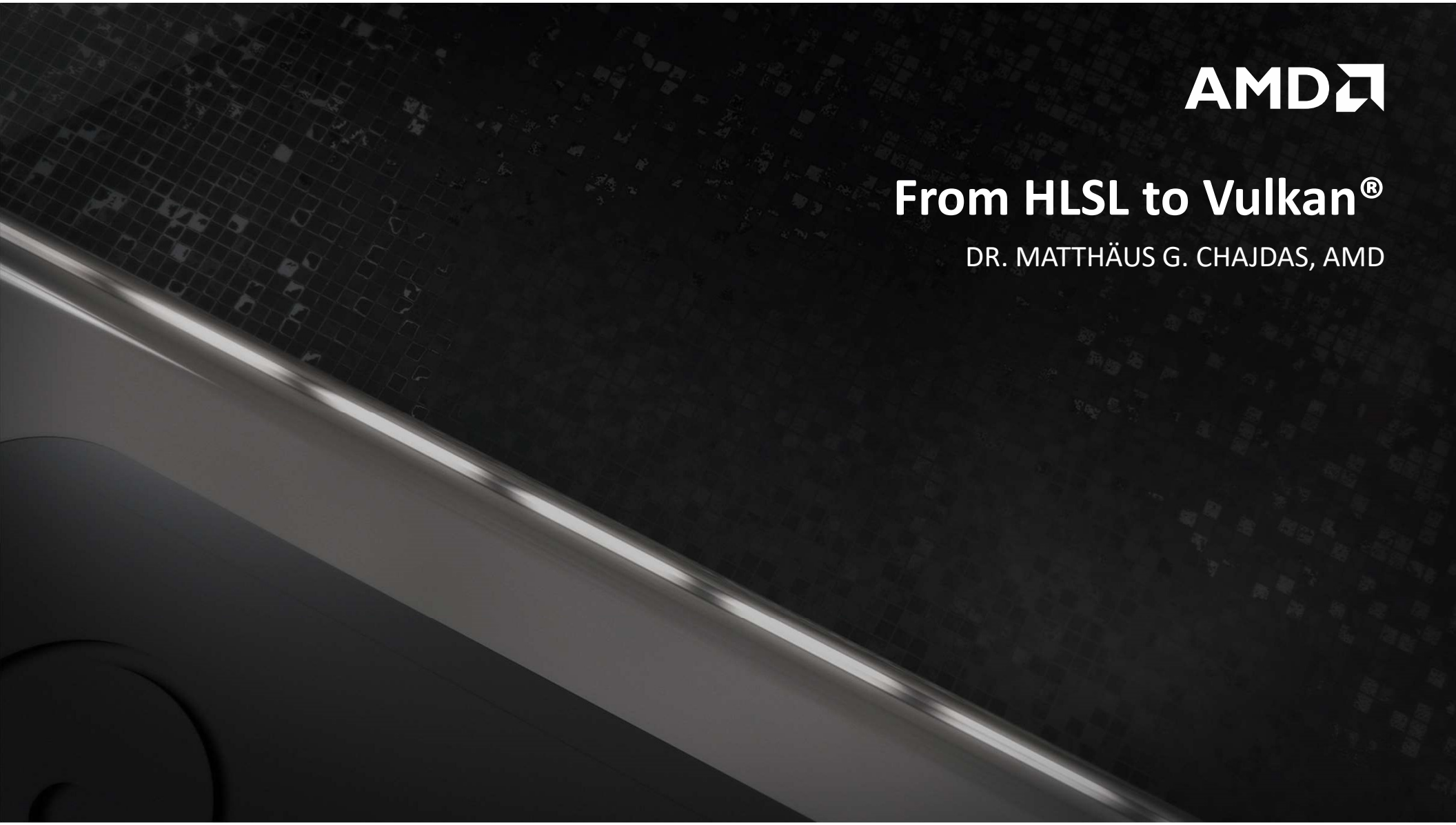
spirv-opt Usage - Indirect through Frontends

- dxc
  - Microsoft Open Source HLSL FE / SPIR-V generator
  - Default Legalization, Optimization
  - Can be disabled (see README)
  - github.com/Microsoft/DirectXShaderCompiler/wiki/SPIR-V-CodeGen
- glslc
  - Wrapper for glslangValidator and SPIRV-Tools
  - Legalization always on for HLSL
  - -Os for size optimization
  - github.com/google/shaderc/tree/master/glslc



Acknowledgements

- Google (David Neto, Steven Perron, Alan Baker, Diego Novillo, John Kessenich, Lei Zhang)
- Valve (Pierre-Loup Griffais, Dan Ginsburg)
- Mesa (Pierre Moreau)
- Roblox (Arseny Kapoulkine)



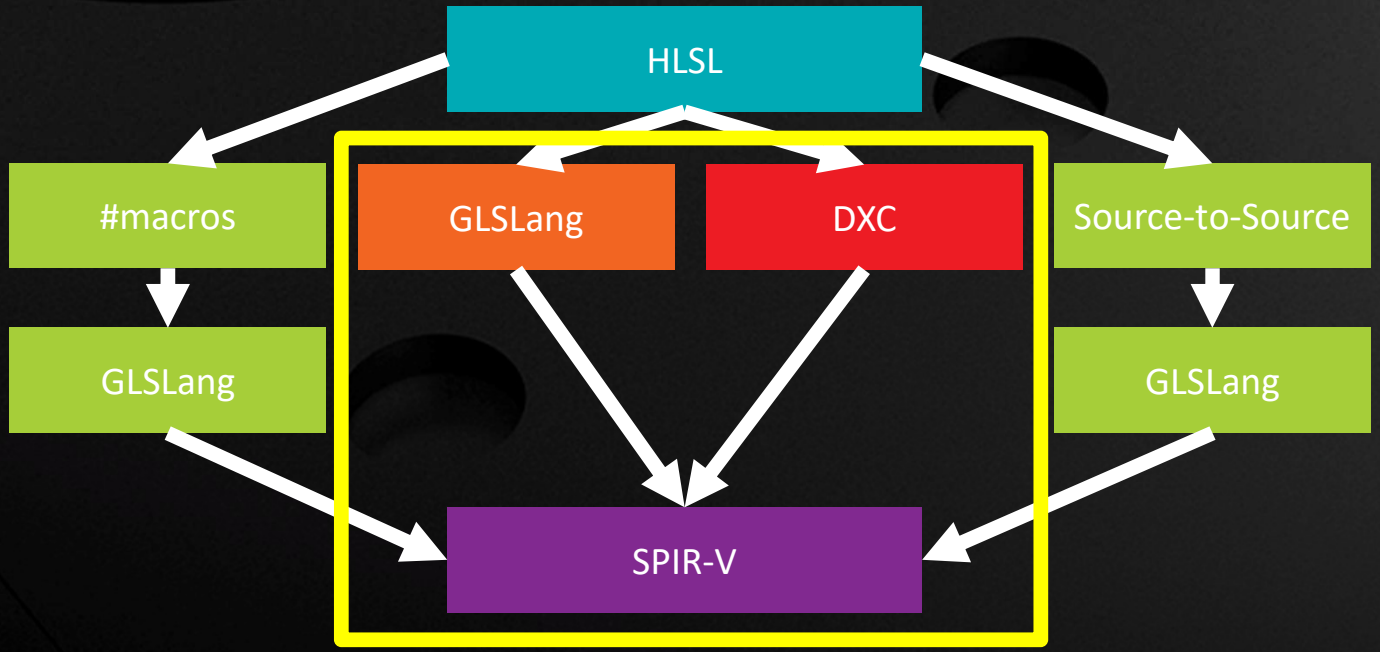
From HLSL to Vulkan®

DR. MATTHÄUS G. CHAJDAS, AMD

STATE OF THE UNION

WHERE ARE WE?

- Most games have large HLSL shader libraries
- Using those with Vulkan requires some work

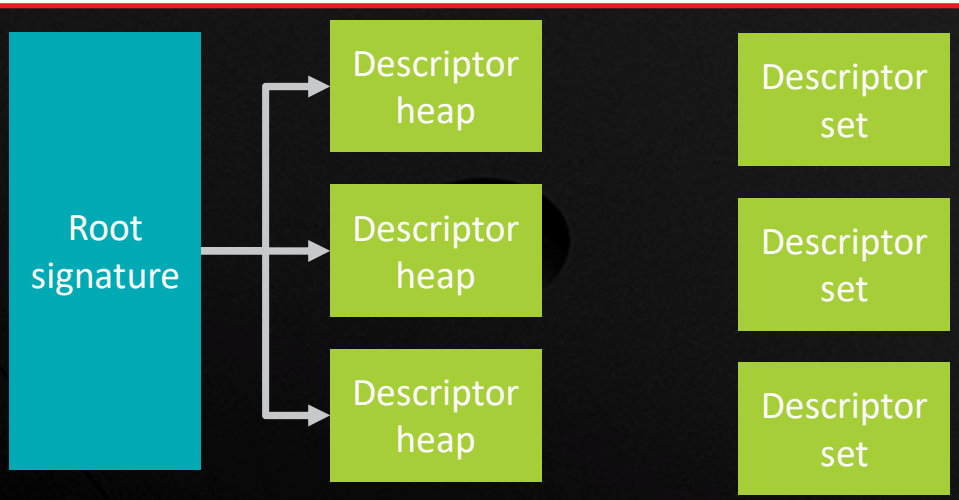


WHAT ARE THE PROBLEMS?

WHY ARE WE DOING THIS? WHY?

- Vulkan has a different binding model from Direct3D 11/12
  - Direct3D 11 is all about slots and named bindings
  - Direct3D 12 has root signatures, typed descriptor heaps etc.
- Vulkan has descriptor sets

```
Texture2D t : register (t0)  ⇔ layout (set = X, binding = Y) texture2D t;
```



OTHER DIFFERENCES

GLSL VS HLSL

- **No 1:1 mapping** of resource types
  - UAV ⇔ Image?
  - Some things have no 100% equivalent, like read-only structured buffers
- **Samplers are not objects** you can pass around
- Bindless is very different
- Descriptor remapping is generally the main problem
  - Partition descriptor ranges?
  - I/O remapper?
  - Engine needs to be somehow aware of both
- Some functionality is **missing** in HLSL
  - Push constants
  - Input attachments
  - Specialization constants

BINDINGS

THE BIG DIFFERENCE

- The big difference is the binding model
- Specific syntax to help you out – attributes "`[[vk::binding]]`" etc.
- Probably want to wrap those in macros so FXC doesn't see this
- This is going to be the majority of your porting effort!

```
struct S {
    float2 f;
};

[[vk::binding(1)]] StructuredBuffer<S> buffer1;
[[vk::binding(3, 2)]] StructuredBuffer<S> buffer3;

[[vk::input_attachment_index(4)]]
Texture2D<float4> attach;

[[vk::constant_id(13)]] const int ci = 11;
[[vk::push_constant]] cbuffer pcBuf { int a; };

[[vk::location(7)]] float4
main([[vk::location(8)]] float4 input: A) : B
{
    return input + attach.Load(float2(0.5));// * a;
}
```

SPIR-V OPT

CLEAN IT UP

AMD

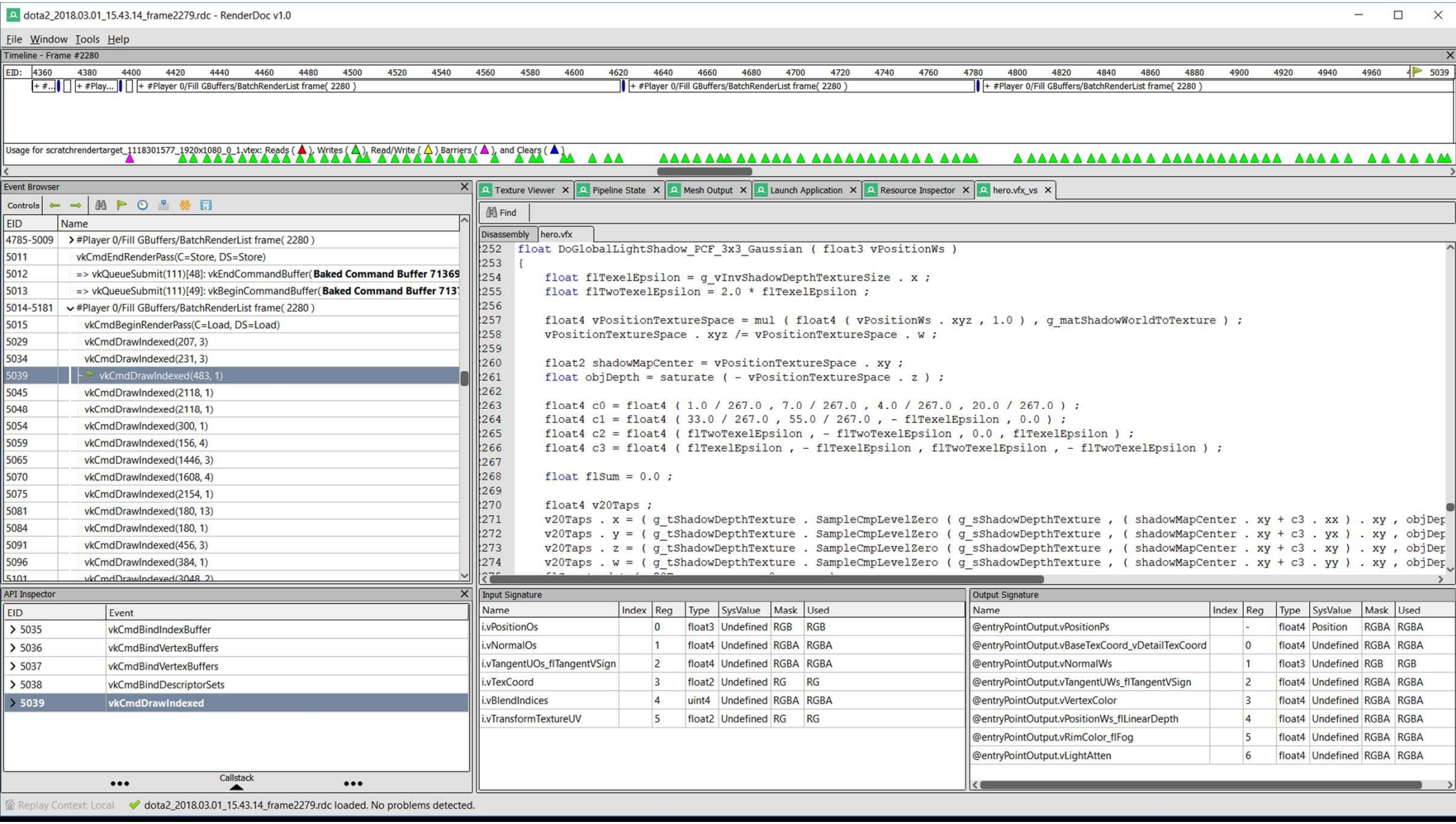
- Your compile pipeline should **end** in SPIR-V Opt
- Required for **legal** SPIR-V from HLSL – mostly for passing around opaque objects (samplers, etc.)
- Various optimizations are available
  - Anything **reducing output size** is generally safe – and also helps the backend
  - Prefer `[unroll]` to forced-unroll
  - Remap identifiers etc. is beneficial
  - Turning on **all options** is not a good idea ☺
- You should also use it for GLSL
- Significant savings on shipping titles on the code size end
  - Still need to apply compression
  - Consider **domain-specific compression** like SMOL-V

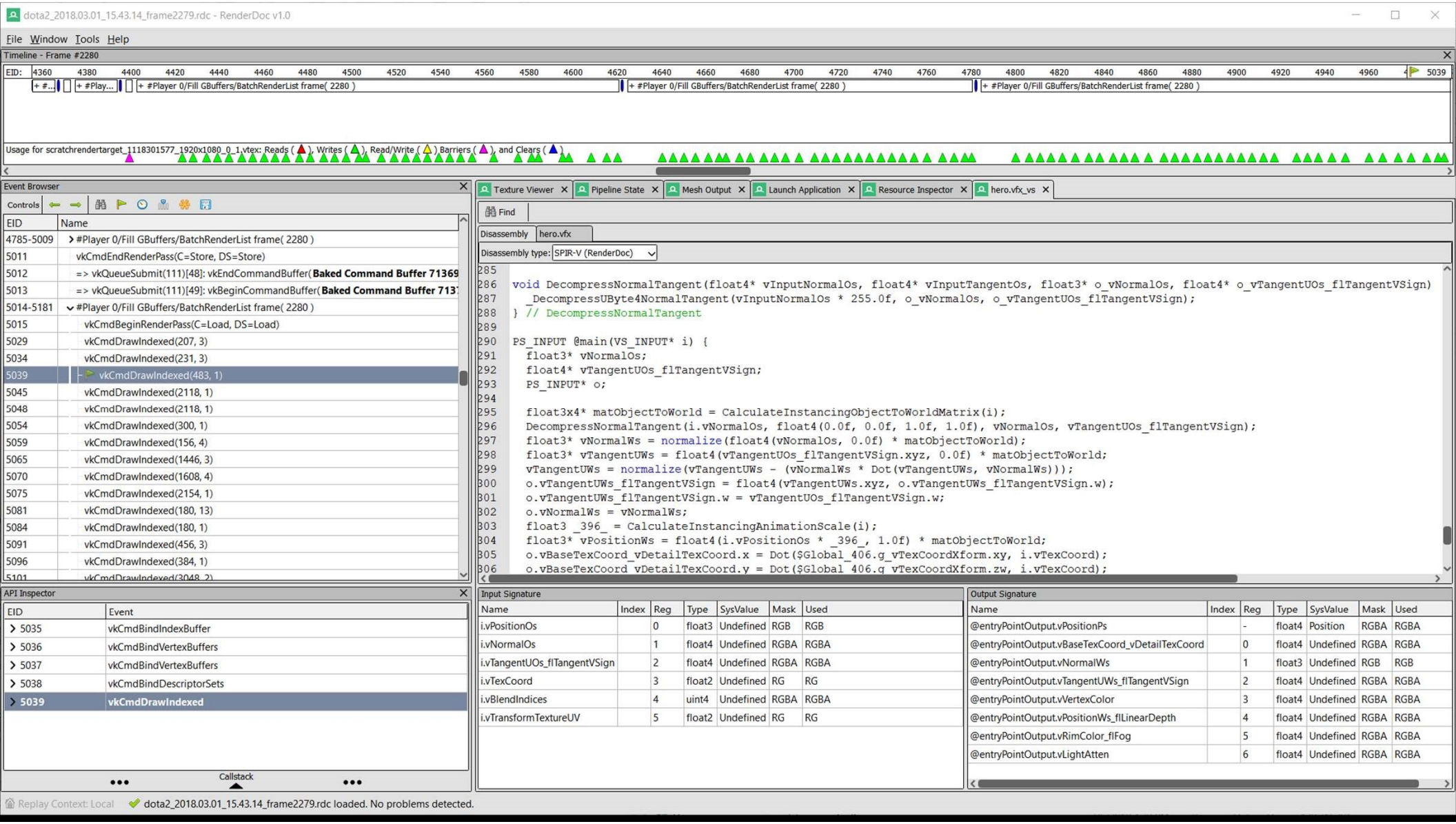
PERF!

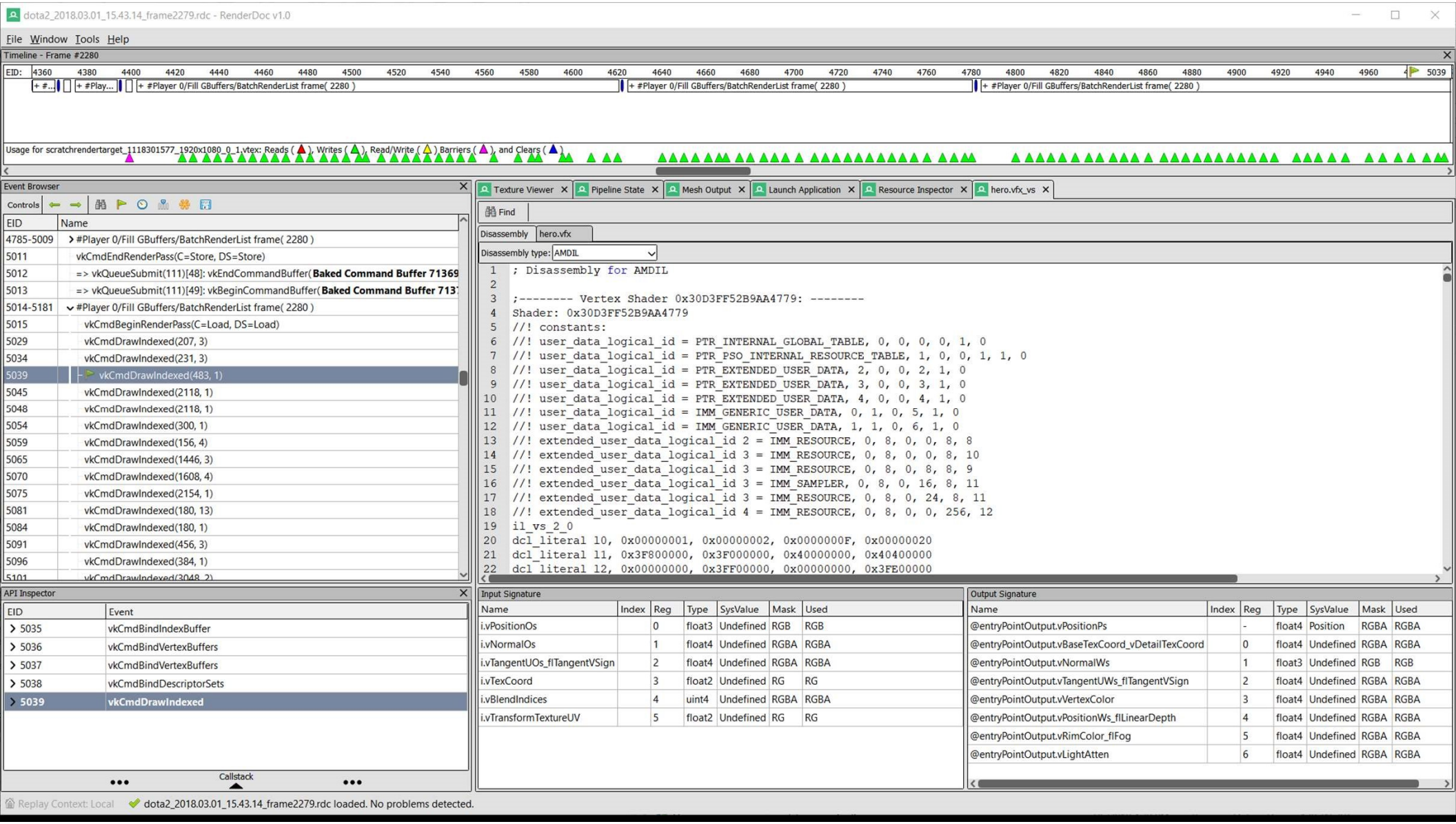
ARE WE FAST YET?

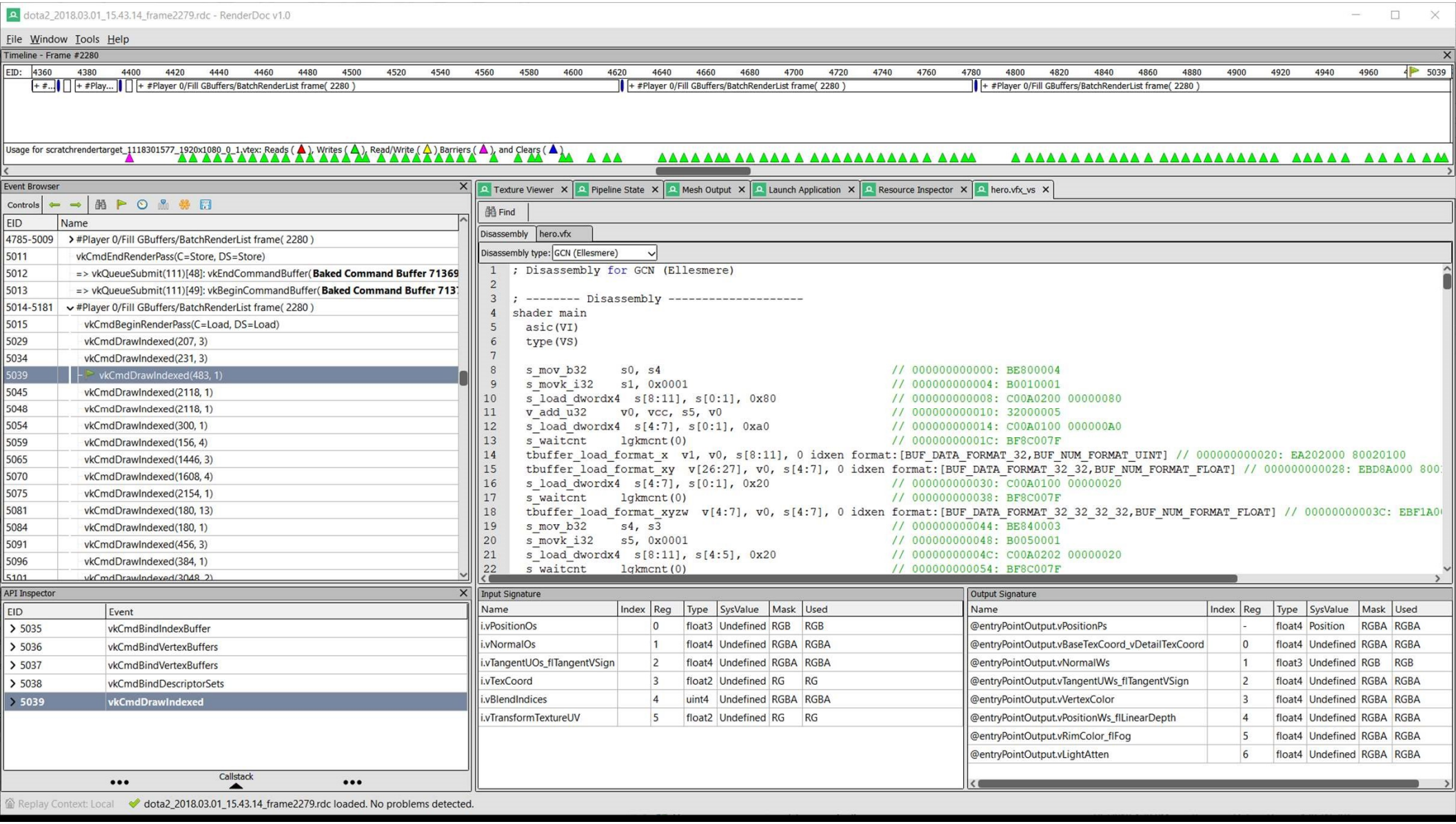
AMD

- Originally, there used to be a significant “Vulkan tax”
  - Compilers not used to seeing SPIR-V
  - GLSLang compile output sometimes rather naïve …
- Be **on the lookout**
  - Temporary/local arrays or arrays in general
  - Function calls – specifically passing large objects around
  - Annotations not getting translated correctly
- Overall, we’re really close these days
  - And if we aren’t, use the tools
  - SPIR-V opt, dis is your friend
  - IHV tools to inspect generated ISA


dota2_2018.03.01_15.43.14_frame2279.rdc - RenderDoc v1.0
File Window Tools Help
Timeline - Frame #2280
Usage for scratchrendertarget_1118301577_1920x1080_0_1.vtex: Reads ( ), Writes ( ), Read/Write ( ) Barriers ( ), and Clears ( )
Event Browser
Controls
EID Name
4785-5009 #Player 0/Fill GBuffers/BatchRenderList frame(2280)
5011 vkCmdEndRenderPass(C=Store, DS=Store)
5012 => vkQueueSubmit(111)[48]: vkEndCommandBuffer(Baked Command Buffer 71369
5013 => vkQueueSubmit(111)[49]: vkBeginCommandBuffer(Baked Command Buffer 713
5014-5181 #Player 0/Fill GBuffers/BatchRenderList frame(2280)
5015 vkCmdBeginRenderPass(C=Load, DS=Load)
5029 vkCmdDrawIndexed(207, 3)
5034 vkCmdDrawIndexed(231, 3)
5039 vkCmdDrawIndexed(483, 1)
5045 vkCmdDrawIndexed(2118, 1)
5048 vkCmdDrawIndexed(2118, 1)
5054 vkCmdDrawIndexed(300, 1)
5059 vkCmdDrawIndexed(156, 4)
5065 vkCmdDrawIndexed(1446, 3)
5070 vkCmdDrawIndexed(1608, 4)
5075 vkCmdDrawIndexed(2154, 1)
5081 vkCmdDrawIndexed(180, 13)
5084 vkCmdDrawIndexed(180, 1)
5091 vkCmdDrawIndexed(456, 3)
5096 vkCmdDrawIndexed(384, 1)
API Inspector
EID Event
5035 vkCmdBindIndexBuffer
5036 vkCmdBindVertexBuffers
5037 vkCmdBindVertexBuffers
5038 vkCmdBindDescriptorSets
5039 vkCmdDrawIndexed
Callstack
Texture Viewer
Pipeline State
Mesh Output
Launch Application
Resource Inspector
hero.vfx_vs
Find
Disassembly
hero.vfx
float DoGlobalLightShadow_PCF_3x3_Gaussian (float3 vPositionWs)
{
float flTexelEpsilon = g_vInvShadowDepthTextureSize . x ;
float flTwoTexelEpsilon = 2.0 * flTexelEpsilon ;
float4 vPositionTextureSpace = mul (float4 (vPositionWs . xyz , 1.0) , g_matShadowWorldToTexture) ;
vPositionTextureSpace . xyz /= vPositionTextureSpace . w ;
float2 shadowMapCenter = vPositionTextureSpace . xy ;
float objDepth = saturate (- vPositionTextureSpace . z) ;
float4 c0 = float4 (1.0 / 267.0 , 7.0 / 267.0 , 4.0 / 267.0 , 20.0 / 267.0) ;
float4 c1 = float4 (33.0 / 267.0 , 55.0 / 267.0 , - flTexelEpsilon , 0.0) ;
float4 c2 = float4 (flTwoTexelEpsilon , - flTwoTexelEpsilon , 0.0 , flTexelEpsilon) ;
float4 c3 = float4 (flTexelEpsilon , - flTexelEpsilon , flTwoTexelEpsilon , - flTwoTexelEpsilon) ;
float flSum = 0.0 ;
float4 v20Taps ;
v20Taps . x = (g_tShadowDepthTexture . SampleCmpLevelZero (g_sShadowDepthTexture , (shadowMapCenter . xy + c3 . xx) . xy , objDep
v20Taps . y = (g_tShadowDepthTexture . SampleCmpLevelZero (g_sShadowDepthTexture , (shadowMapCenter . xy + c3 . yx) . xy , objDep
v20Taps . z = (g_tShadowDepthTexture . SampleCmpLevelZero (g_sShadowDepthTexture , (shadowMapCenter . xy + c3 . xy) . xy , objDep
v20Taps . w = (g_tShadowDepthTexture . SampleCmpLevelZero (g_sShadowDepthTexture , (shadowMapCenter . xy + c3 . yy) . xy , objDep
Input Signature
Name Index Reg Type SysValue Mask Used
i.vPositionOs 0 float3 Undefined RGB RGB
i.vNormalOs 1 float4 Undefined RGBA RGBA
i.vTangentUOs_flTangentVSign 2 float4 Undefined RGBA RGBA
i.vTexCoord 3 float2 Undefined RG RG
i.vBlendIndices 4 uint4 Undefined RGBA RGBA
i.vTransformTextureUV 5 float2 Undefined RG RG
Output Signature
Name Index Reg Type SysValue Mask Used
@entryPointOutput.vPositionPs - float4 Position RGBA RGBA
@entryPointOutput.vBaseTexCoord_vDetailTexCoord 0 float4 Undefined RGBA RGBA
@entryPointOutput.vNormalWs 1 float3 Undefined RGB RGB
@entryPointOutput.vTangentUWs_flTangentVSign 2 float4 Undefined RGBA RGBA
@entryPointOutput.vVertexColor 3 float4 Undefined RGBA RGBA
@entryPointOutput.vPositionWs_flLinearDepth 4 float4 Undefined RGBA RGBA
@entryPointOutput.vRimColor_flFog 5 float4 Undefined RGBA RGBA
@entryPointOutput.vLightAtten 6 float4 Undefined RGBA RGBA
Replay Context: Local
dota2_2018.03.01_15.43.14_frame2279.rdc loaded. No problems detected.


dota2_2018.03.01_15.43.14_frame2279.rdc - RenderDoc v1.0
File Window Tools Help
Timeline - Frame #2280
Usage for scratchrendertarget_1118301577_1920x1080_0_1.vtex: Reads ( ), Writes ( ), Read/Write ( ) Barriers ( ), and Clears ( )
Event Browser
Controls
EID Name
4785-5009 #Player 0/Fill GBuffers/BatchRenderList frame(2280)
5011 vkCmdEndRenderPass(C=Store, DS=Store)
5012 => vkQueueSubmit(111)[48]: vkEndCommandBuffer(Baked Command Buffer 71369
5013 => vkQueueSubmit(111)[49]: vkBeginCommandBuffer(Baked Command Buffer 713
5014-5181 #Player 0/Fill GBuffers/BatchRenderList frame(2280)
5015 vkCmdBeginRenderPass(C=Load, DS=Load)
5029 vkCmdDrawIndexed(207, 3)
5034 vkCmdDrawIndexed(231, 3)
5039 vkCmdDrawIndexed(483, 1)
5045 vkCmdDrawIndexed(2118, 1)
5048 vkCmdDrawIndexed(2118, 1)
5054 vkCmdDrawIndexed(300, 1)
5059 vkCmdDrawIndexed(156, 4)
5065 vkCmdDrawIndexed(1446, 3)
5070 vkCmdDrawIndexed(1608, 4)
5075 vkCmdDrawIndexed(2154, 1)
5081 vkCmdDrawIndexed(180, 13)
5084 vkCmdDrawIndexed(180, 1)
5091 vkCmdDrawIndexed(456, 3)
5096 vkCmdDrawIndexed(384, 1)
API Inspector
EID Event
5035 vkCmdBindIndexBuffer
5036 vkCmdBindVertexBuffers
5037 vkCmdBindVertexBuffers
5038 vkCmdBindDescriptorSets
5039 vkCmdDrawIndexed
Callstack
Texture Viewer
Pipeline State
Mesh Output
Launch Application
Resource Inspector
hero.vfx_vs
Find
Disassembly
hero.vfx
Disassembly type: SPIR-V (RenderDoc)
285
286 void DecompressNormalTangent(float4* vInputNormalOs, float4* vInputTangentOs, float3* o_vNormalOs, float4* o_vTangentUOs_flTangentVSign)
287 _DecompressUByte4NormalTangent(vInputNormalOs * 255.0f, o_vNormalOs, o_vTangentUOs_flTangentVSign);
288 } // DecompressNormalTangent
289
290 PS_INPUT @main(VS_INPUT* i) {
291 float3* vNormalOs;
292 float4* vTangentUOs_flTangentVSign;
293 PS_INPUT* o;
294
295 float3x4* matObjectToWorld = CalculateInstancingObjectToWorldMatrix(i);
296 DecompressNormalTangent(i.vNormalOs, float4(0.0f, 0.0f, 1.0f, 1.0f), vNormalOs, vTangentUOs_flTangentVSign);
297 float3* vNormalWs = normalize(float4(vNormalOs, 0.0f) * matObjectToWorld);
298 float3* vTangentUWs = float4(vTangentUOs_flTangentVSign.xyz, 0.0f) * matObjectToWorld;
299 vTangentUWs = normalize(vTangentUWs - (vNormalWs * Dot(vTangentUWs, vNormalWs)));
300 o.vTangentUWs_flTangentVSign = float4(vTangentUWs.xyz, o.vTangentUWs_flTangentVSign.w);
301 o.vTangentUWs_flTangentVSign.w = vTangentUOs_flTangentVSign.w;
302 o.vNormalWs = vNormalWs;
303 float3 _396_ = CalculateInstancingAnimationScale(i);
304 float3* vPositionWs = float4(i.vPositionOs * _396_, 1.0f) * matObjectToWorld;
305 o.vBaseTexCoord_vDetailTexCoord.x = Dot($Global_406.g_vTexCoordXform.xy, i.vTexCoord);
306 o.vBaseTexCoord_vDetailTexCoord.y = Dot($Global_406.g_vTexCoordXform.zw, i.vTexCoord);
Input Signature
Name Index Reg Type SysValue Mask Used
i.vPositionOs 0 float3 Undefined RGB RGB
i.vNormalOs 1 float4 Undefined RGBA RGBA
i.vTangentUOs_flTangentVSign 2 float4 Undefined RGBA RGBA
i.vTexCoord 3 float2 Undefined RG RG
i.vBlendIndices 4 uint4 Undefined RGBA RGBA
i.vTransformTextureUV 5 float2 Undefined RG RG
Output Signature
Name Index Reg Type SysValue Mask Used
@entryPointOutput.vPositionPs - float4 Position RGBA RGBA
@entryPointOutput.vBaseTexCoord_vDetailTexCoord 0 float4 Undefined RGBA RGBA
@entryPointOutput.vNormalWs 1 float3 Undefined RGB RGB
@entryPointOutput.vTangentUWs_flTangentVSign 2 float4 Undefined RGBA RGBA
@entryPointOutput.vVertexColor 3 float4 Undefined RGBA RGBA
@entryPointOutput.vPositionWs_flLinearDepth 4 float4 Undefined RGBA RGBA
@entryPointOutput.vRimColor_flFog 5 float4 Undefined RGBA RGBA
@entryPointOutput.vLightAtten 6 float4 Undefined RGBA RGBA
Replay Context: Local
dota2_2018.03.01_15.43.14_frame2279.rdc loaded. No problems detected.

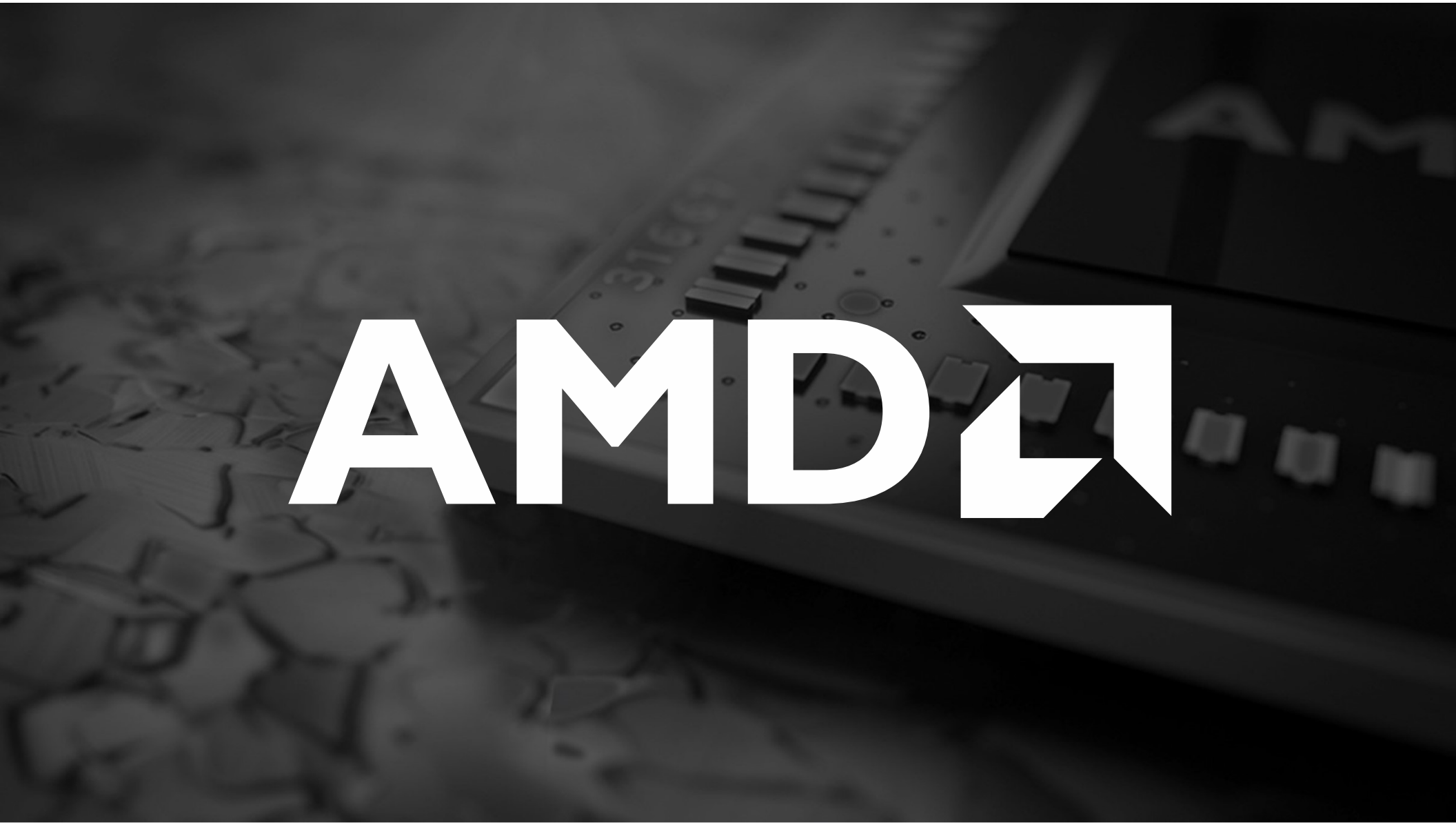

dota2_2018.03.01_15.43.14_frame2279.rdc - RenderDoc v1.0
File Window Tools Help
Timeline - Frame #2280
Usage for scratchrendertarget_1118301577_1920x1080_0_1.vtex: Reads ( ), Writes ( ), Read/Write ( ) Barriers ( ), and Clears ( )
Event Browser
Controls
EID Name
4785-5009 #Player 0/Fill GBuffers/BatchRenderList frame(2280)
5011 vkCmdEndRenderPass(C=Store, DS=Store)
5012 => vkQueueSubmit(111)[48]: vkEndCommandBuffer(Baked Command Buffer 71369
5013 => vkQueueSubmit(111)[49]: vkBeginCommandBuffer(Baked Command Buffer 713
5014-5181 #Player 0/Fill GBuffers/BatchRenderList frame(2280)
5015 vkCmdBeginRenderPass(C=Load, DS=Load)
5029 vkCmdDrawIndexed(207, 3)
5034 vkCmdDrawIndexed(231, 3)
5039 vkCmdDrawIndexed(483, 1)
5045 vkCmdDrawIndexed(2118, 1)
5048 vkCmdDrawIndexed(2118, 1)
5054 vkCmdDrawIndexed(300, 1)
5059 vkCmdDrawIndexed(156, 4)
5065 vkCmdDrawIndexed(1446, 3)
5070 vkCmdDrawIndexed(1608, 4)
5075 vkCmdDrawIndexed(2154, 1)
5081 vkCmdDrawIndexed(180, 13)
5084 vkCmdDrawIndexed(180, 1)
5091 vkCmdDrawIndexed(456, 3)
5096 vkCmdDrawIndexed(384, 1)
API Inspector
EID Event
5035 vkCmdBindIndexBuffer
5036 vkCmdBindVertexBuffers
5037 vkCmdBindVertexBuffers
5038 vkCmdBindDescriptorSets
5039 vkCmdDrawIndexed
Callstack
Texture Viewer
Pipeline State
Mesh Output
Launch Application
Resource Inspector
hero.vfx_vs
Find
Disassembly
hero.vfx
Disassembly type: AMDIL
; Disassembly for AMDIL
;-------- Vertex Shader 0x30D3FF52B9AA4779: --------
Shader: 0x30D3FF52B9AA4779
//! constants:
//! user_data_logical_id = PTR_INTERNAL_GLOBAL_TABLE, 0, 0, 0, 0, 1, 0
//! user_data_logical_id = PTR_PSO_INTERNAL_RESOURCE_TABLE, 1, 0, 0, 1, 1, 0
//! user_data_logical_id = PTR_EXTENDED_USER_DATA, 2, 0, 0, 2, 1, 0
//! user_data_logical_id = PTR_EXTENDED_USER_DATA, 3, 0, 0, 3, 1, 0
//! user_data_logical_id = PTR_EXTENDED_USER_DATA, 4, 0, 0, 4, 1, 0
//! user_data_logical_id = IMM_GENERIC_USER_DATA, 0, 1, 0, 5, 1, 0
//! user_data_logical_id = IMM_GENERIC_USER_DATA, 1, 1, 0, 6, 1, 0
//! extended_user_data_logical_id 2 = IMM_RESOURCE, 0, 8, 0, 0, 8, 8
//! extended_user_data_logical_id 3 = IMM_RESOURCE, 0, 8, 0, 0, 8, 10
//! extended_user_data_logical_id 3 = IMM_RESOURCE, 0, 8, 0, 8, 8, 9
//! extended_user_data_logical_id 3 = IMM_SAMPLER, 0, 8, 0, 16, 8, 11
//! extended_user_data_logical_id 3 = IMM_RESOURCE, 0, 8, 0, 24, 8, 11
//! extended_user_data_logical_id 4 = IMM_RESOURCE, 0, 8, 0, 0, 256, 12
il_vs_2_0
dcl_literal l0, 0x00000001, 0x00000002, 0x0000000F, 0x00000020
dcl_literal l1, 0x3F800000, 0x3F000000, 0x40000000, 0x40400000
dcl_literal l2, 0x00000000, 0x3FF00000, 0x00000000, 0x3FE00000
Input Signature
Name Index Reg Type SysValue Mask Used
i.vPositionOs 0 float3 Undefined RGB RGB
i.vNormalOs 1 float4 Undefined RGBA RGBA
i.vTangentUOs_flTangentVSign 2 float4 Undefined RGBA RGBA
i.vTexCoord 3 float2 Undefined RG RG
i.vBlendIndices 4 uint4 Undefined RGBA RGBA
i.vTransformTextureUV 5 float2 Undefined RG RG
Output Signature
Name Index Reg Type SysValue Mask Used
@entryPointOutput.vPositionPs - float4 Position RGBA RGBA
@entryPointOutput.vBaseTexCoord_vDetailTexCoord 0 float4 Undefined RGBA RGBA
@entryPointOutput.vNormalWs 1 float3 Undefined RGB RGB
@entryPointOutput.vTangentUWs_flTangentVSign 2 float4 Undefined RGBA RGBA
@entryPointOutput.vVertexColor 3 float4 Undefined RGBA RGBA
@entryPointOutput.vPositionWs_flLinearDepth 4 float4 Undefined RGBA RGBA
@entryPointOutput.vRimColor_flFog 5 float4 Undefined RGBA RGBA
@entryPointOutput.vLightAtten 6 float4 Undefined RGBA RGBA
Replay Context: Local
dota2_2018.03.01_15.43.14_frame2279.rdc loaded. No problems detected.


dota2_2018.03.01_15.43.14_frame2279.rdc - RenderDoc v1.0
File Window Tools Help
Timeline - Frame #2280
EID: 4360 4380 4400 4420 4440 4460 4480 4500 4520 4540 4560 4580 4600 4620 4640 4660 4680 4700 4720 4740 4760 4780 4800 4820 4840 4860 4880 4900 4920 4940 4960 5039
+ #Player 0/Fill GBuffers/BatchRenderList frame(2280)
+ #Player 0/Fill GBuffers/BatchRenderList frame(2280)
+ #Player 0/Fill GBuffers/BatchRenderList frame(2280)
Usage for scratchrendertarget_1118301577_1920x1080_0_1.vtex: Reads ( ), Writes ( ), Read/Write ( ) Barriers ( ), and Clears ( )
Event Browser
Controls
EID Name
4785-5009 #Player 0/Fill GBuffers/BatchRenderList frame(2280)
5011 vkCmdEndRenderPass(C=Store, DS=Store)
5012 => vkQueueSubmit(111)[48]: vkEndCommandBuffer(Baked Command Buffer 71369
5013 => vkQueueSubmit(111)[49]: vkBeginCommandBuffer(Baked Command Buffer 713
5014-5181 #Player 0/Fill GBuffers/BatchRenderList frame(2280)
5015 vkCmdBeginRenderPass(C=Load, DS=Load)
5029 vkCmdDrawIndexed(207, 3)
5034 vkCmdDrawIndexed(231, 3)
5039 vkCmdDrawIndexed(483, 1)
5045 vkCmdDrawIndexed(2118, 1)
5048 vkCmdDrawIndexed(2118, 1)
5054 vkCmdDrawIndexed(300, 1)
5059 vkCmdDrawIndexed(156, 4)
5065 vkCmdDrawIndexed(1446, 3)
5070 vkCmdDrawIndexed(1608, 4)
5075 vkCmdDrawIndexed(2154, 1)
5081 vkCmdDrawIndexed(180, 13)
5084 vkCmdDrawIndexed(180, 1)
5091 vkCmdDrawIndexed(456, 3)
5096 vkCmdDrawIndexed(384, 1)
5101 vkCmdDrawIndexed(3048, 2)
API Inspector
EID Event
5035 vkCmdBindIndexBuffer
5036 vkCmdBindVertexBuffers
5037 vkCmdBindVertexBuffers
5038 vkCmdBindDescriptorSets
5039 vkCmdDrawIndexed
Callstack
Texture Viewer
Pipeline State
Mesh Output
Launch Application
Resource Inspector
hero.vfx_vs
Find
Disassembly
hero.vfx
Disassembly type: GCN (Ellesmere)
1 ; Disassembly for GCN (Ellesmere)
2
3 ; -------- Disassembly --------------------
4 shader main
5 asic(VI)
6 type(VS)
7
8 s_mov_b32 s0, s4 // 000000000000: BE800004
9 s_movk_i32 s1, 0x0001 // 000000000004: B0010001
10 s_load_dwordx4 s[8:11], s[0:1], 0x80 // 000000000008: C00A0200 00000080
11 v_add_u32 v0, vcc, s5, v0 // 000000000010: 32000005
12 s_load_dwordx4 s[4:7], s[0:1], 0xa0 // 000000000014: C00A0100 000000A0
13 s_waitcnt lgkmcnt(0) // 00000000001C: BF8C007F
14 tbuffer_load_format_x v1, v0, s[8:11], 0 idxen format:[BUF_DATA_FORMAT_32,BUF_NUM_FORMAT_UINT] // 000000000020: EA202000 80020100
15 tbuffer_load_format_xy v[26:27], v0, s[4:7], 0 idxen format:[BUF_DATA_FORMAT_32_32,BUF_NUM_FORMAT_FLOAT] // 000000000028: EBD8A000 800
16 s_load_dwordx4 s[4:7], s[0:1], 0x20 // 000000000030: C00A0100 00000020
17 s_waitcnt lgkmcnt(0) // 000000000038: BF8C007F
18 tbuffer_load_format_xyzw v[4:7], v0, s[4:7], 0 idxen format:[BUF_DATA_FORMAT_32_32_32_32,BUF_NUM_FORMAT_FLOAT] // 00000000003C: EBF1A0
19 s_mov_b32 s4, s3 // 000000000044: BE840003
20 s_movk_i32 s5, 0x0001 // 000000000048: B0050001
21 s_load_dwordx4 s[8:11], s[4:5], 0x20 // 00000000004C: C00A0202 00000020
22 s_waitcnt lgkmcnt(0) // 000000000054: BF8C007F
Input Signature
Name Index Reg Type SysValue Mask Used
i.vPositionOs 0 float3 Undefined RGB RGB
i.vNormalOs 1 float4 Undefined RGBA RGBA
i.vTangentUOs_flTangentVSign 2 float4 Undefined RGBA RGBA
i.vTexCoord 3 float2 Undefined RG RG
i.vBlendIndices 4 uint4 Undefined RGBA RGBA
i.vTransformTextureUV 5 float2 Undefined RG RG
Output Signature
Name Index Reg Type SysValue Mask Used
@entryPointOutput.vPositionPs - float4 Position RGBA RGBA
@entryPointOutput.vBaseTexCoord_vDetailTexCoord 0 float4 Undefined RGBA RGBA
@entryPointOutput.vNormalWs 1 float3 Undefined RGB RGB
@entryPointOutput.vTangentUWs_flTangentVSign 2 float4 Undefined RGBA RGBA
@entryPointOutput.vVertexColor 3 float4 Undefined RGBA RGBA
@entryPointOutput.vPositionWs_flLinearDepth 4 float4 Undefined RGBA RGBA
@entryPointOutput.vRimColor_flFog 5 float4 Undefined RGBA RGBA
@entryPointOutput.vLightAtten 6 float4 Undefined RGBA RGBA
Replay Context: Local
dota2_2018.03.01_15.43.14_frame2279.rdc loaded. No problems detected.

SUMMARY

WHERE ARE WE?

- Games **are shipping** on Vulkan with tons of HLSL source
- **Single HLSL source** for both Vulkan and D3D is possible
- Going forward, you can use DXC supporting both APIs – more on this in a minute!


AMD

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

DXC / Spiregg

Hai Nguyen / Google
March 2018

Overview

- **Brief History**
  - There and back again
- **Development**
  - Compilation
  - Legalization
  - Optimization
- **What's Here**
  - SM6.0 Wave Ops
  - Semantics
  - Reflection
- **What's Coming**
  - Descriptor Indexing
  - Extensions
- **Credits**
  - Appreciating the blood, sweat, and tears

Brief History

- **There…**
  - David Neto (Google) initiated the Spiregg project
    - Google met with Microsoft DXC team in early 2017
  - HLSL is a language without a spec
    - "I did not know that was legal…" – Baldur Karlsson (RenderDoc)
    - Microsoft plans to evolve the HLSL rapidly once DXC is stable
    - Keeping up with parser could prove difficult…so leverage it!
- **…and back again**
  - Spiregg at Google contributes and maintains a SPIR-V backend for DXC
    - Open source under LLVM license and hosted on Github
  - Spiregg leaned on glslang's progress to get started
    - Both projects collaborate to maintain parity
  - Spiregg works closely with SPIRV-Tools/spirv-opt team
    - DXC/SPIRV leans on spirv-opt for legalization
  - Outreach to community and IHVs has provided valuable feedback

Development

- **Compilation**
  - Parsing HLSL to generalized SPIR-V
  - ***Front end*** process that happens in DXC
- **Legalization**
  - Transforming generalized SPIR-V to Vulkan dialect
  - ***Back end*** process that happens in SPIRV-Tools
    - spirv-opt transform passes
  - See Greg's talk for more details
- **Optimization**
  - Transforms Vulkan SPIR-V to be more performant
  - Controversial transform
    - Loop unrolling
      - Must be done for legalization, will follow up about performance
    - Will follow up at SIGGRAPH

Development

- **Spiregg has been very busy!**
  - Lei Zhang (antiagainst)
  - Ehsan Nasiri (ehsannas)

Legalization

- **We are aware that some generated SPIR-V is problematic for consumption**
  - Complex HLSL opaque object resolution
    - Structs within structs containing opaque objects
  - First time legalization has been attempted for HLSL in Vulkan
    - We're learning very valuable lessons
- **Working with IHVs to address this as quickly as possible**
  - Bug fixes to DXC
  - Updated drivers

What's Here

- **Full SM 5.1 Support**
  - Bugs (and features not in Vulkan) not withstanding
    - Please bugs report on Github!
  - Let us know if we're missing anything!
- **SM 6.0 Wave Ops**
  - Landed with Vulkan 1.1
- **Other Highlights**
  - Global variable collected under $Globals cbuffer
    - Working on support to assing $Globals to a specific register
  - SPV_KHR_shader_draw_parameters implemented to explore extension workflow
- **Semantics and Counter Buffers!**
  - Two SPIR-V Extensions
  - One upcoming Vulkan Extension to support SPIR-V Extensions
- **Reflection**
  - Reflecting SPIR-V data at runtime

What's Here: Semantics and Counter Buffers

- **Two SPIR-V extensions**
  - SPV_GOOGLE_decorate_string
    - **OpDecorateStringGOOGLE** - decorates variable with string
    - **OpMemberDecorateStringGOOGLE** - decorate struct member with string
  - SPV_GOOGLE_hlsl_functionality1
    - **HlslCounterBufferGOOGLE** - link a counter buffer to a UAV resource that has an associated counter
    - **HlslSemanticGOOGLE** - decorate an input or output variable id with a string representing semantic as defined in the HLSL source
- **One Vulkan extension**
  - WIP
- **Opt-in Feature for DXC and glslang**
  - -fspv-reflect for DXC
  - -hlsl_functionality1 for glslang

What's Here: Semantics and Counter Buffers

C:\code\hai\SPIRV-Reflect\examples\sample.hlsl - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

sample.hlsl

```
24  ConsumeStructuredBuffer<Data> MyBufferIn : register(u3, space2);
25  AppendStructuredBuffer<Data> MyBufferOut : register(u4, space2);
26
27  struct PSInput {
28    float4  Position  : SV_POSITION;
29    float3  Normal    : NORMAL;
30    float3  Color     : COLOR;
31    float   Alpha     : OPACITY;
32    float4  Scaling   : SCALE;
33    float2  TexCoord0 : TEXCOORD0;
34    float2  TexCoord1 : TEXCOORD1;
35    float2  TexCoord2 : TEXCOORD2;
36  };
37
38  struct PSOutput {
39    float4  oColor0 : SV_TARGET0;
40    float4  oColor1 : SV_TARGET1;
41    float4  oColor2 : SV_TARGET2;
42    float4  oColor3 : SV_TARGET3;
43    float4  oColor4 : SV_TARGET4;
44    float4  oColor5 : SV_TARGET5;
45    float4  oColor6 : SV_TARGET6;
46    float4  oColor7 : SV_TARGET7;
47  };
48
```

Line 17, Column 1

MINGW64:/c/code/hai/SPIRV-Reflect/examples

```
hai@DESKTOP-O1RMH1F MINGW64 /c/code/hai/SPIRV-Reflect/examples (master)
$ spirv-dis --raw-id sample_semantics.spv | grep GOOGLE
               OpExtension "SPV_GOOGLE_hlsl_functionality1"
               OpExtension "SPV_GOOGLE_decorate_string"
               OpDecorateId %23 HlslCounterBufferGOOGLE %25
               OpDecorateId %26 HlslCounterBufferGOOGLE %27
               OpDecorateStringGOOGLE %2 HlslSemanticGOOGLE "SV_POSITION"
               OpDecorateStringGOOGLE %3 HlslSemanticGOOGLE "NORMAL"
               OpDecorateStringGOOGLE %4 HlslSemanticGOOGLE "COLOR"
               OpDecorateStringGOOGLE %5 HlslSemanticGOOGLE "OPACITY"
               OpDecorateStringGOOGLE %6 HlslSemanticGOOGLE "SCALE"
               OpDecorateStringGOOGLE %7 HlslSemanticGOOGLE "TEXCOORD0"
               OpDecorateStringGOOGLE %8 HlslSemanticGOOGLE "TEXCOORD1"
               OpDecorateStringGOOGLE %9 HlslSemanticGOOGLE "TEXCOORD2"
               OpDecorateStringGOOGLE %10 HlslSemanticGOOGLE "SV_TARGET0"
               OpDecorateStringGOOGLE %11 HlslSemanticGOOGLE "SV_TARGET1"
               OpDecorateStringGOOGLE %12 HlslSemanticGOOGLE "SV_TARGET2"
               OpDecorateStringGOOGLE %13 HlslSemanticGOOGLE "SV_TARGET3"
               OpDecorateStringGOOGLE %14 HlslSemanticGOOGLE "SV_TARGET4"
               OpDecorateStringGOOGLE %15 HlslSemanticGOOGLE "SV_TARGET5"
               OpDecorateStringGOOGLE %16 HlslSemanticGOOGLE "SV_TARGET6"
               OpDecorateStringGOOGLE %17 HlslSemanticGOOGLE "SV_TARGET7"

hai@DESKTOP-O1RMH1F MINGW64 /c/code/hai/SPIRV-Reflect/examples (master)
$
```

What's Here: Semantics and Counter Buffers

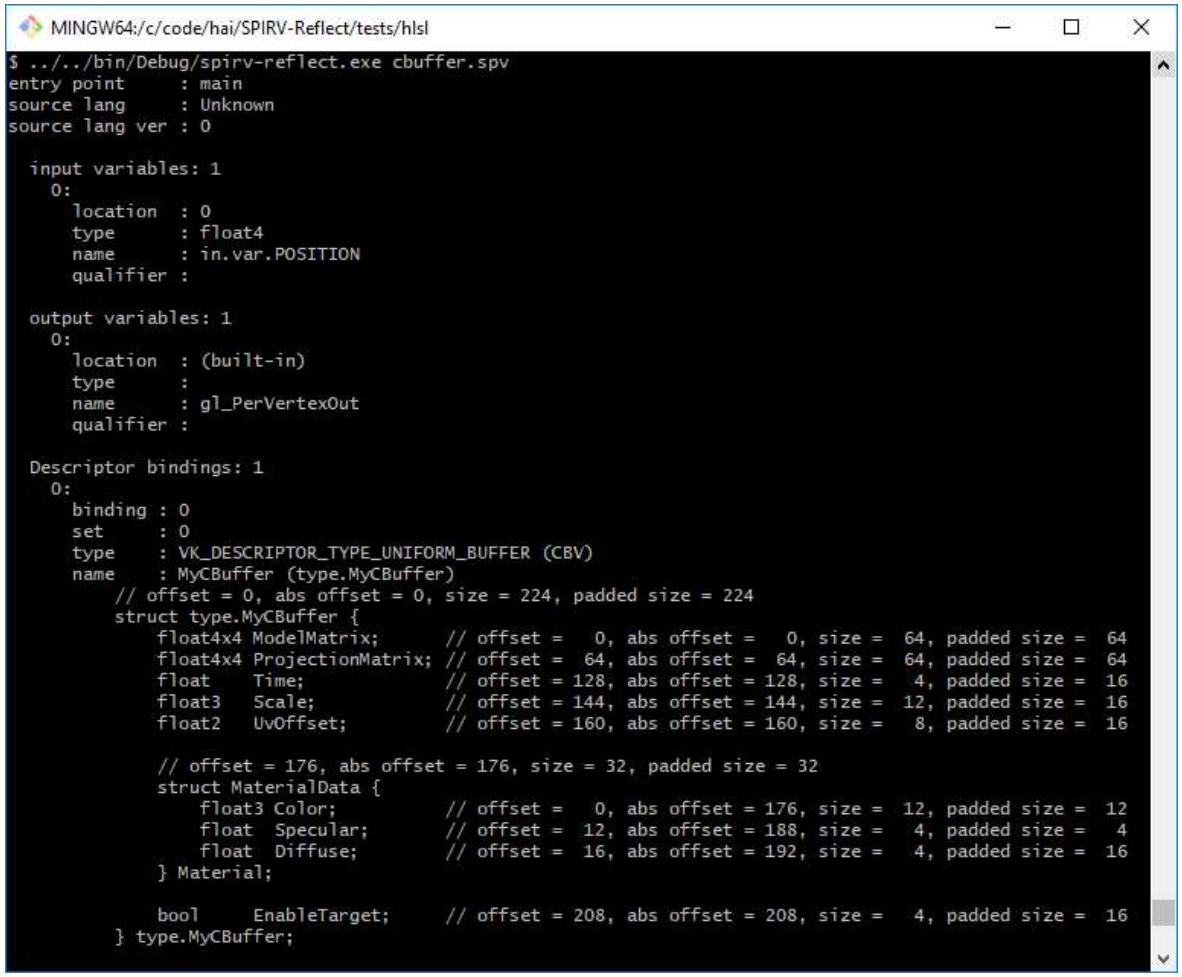
- **There's just one thing…**
  - New SPIR-V opcodes and decorations not yet consumable by drivers
  - Must be stripped before handing off the SPIR-V to the driver
- **Fear not…batteries are included!**
  - SPIRV-Reflect repo will have copy/paste snipped to strip reflection data
  - What's SPIRV-Reflect?

What's Here: Reflection

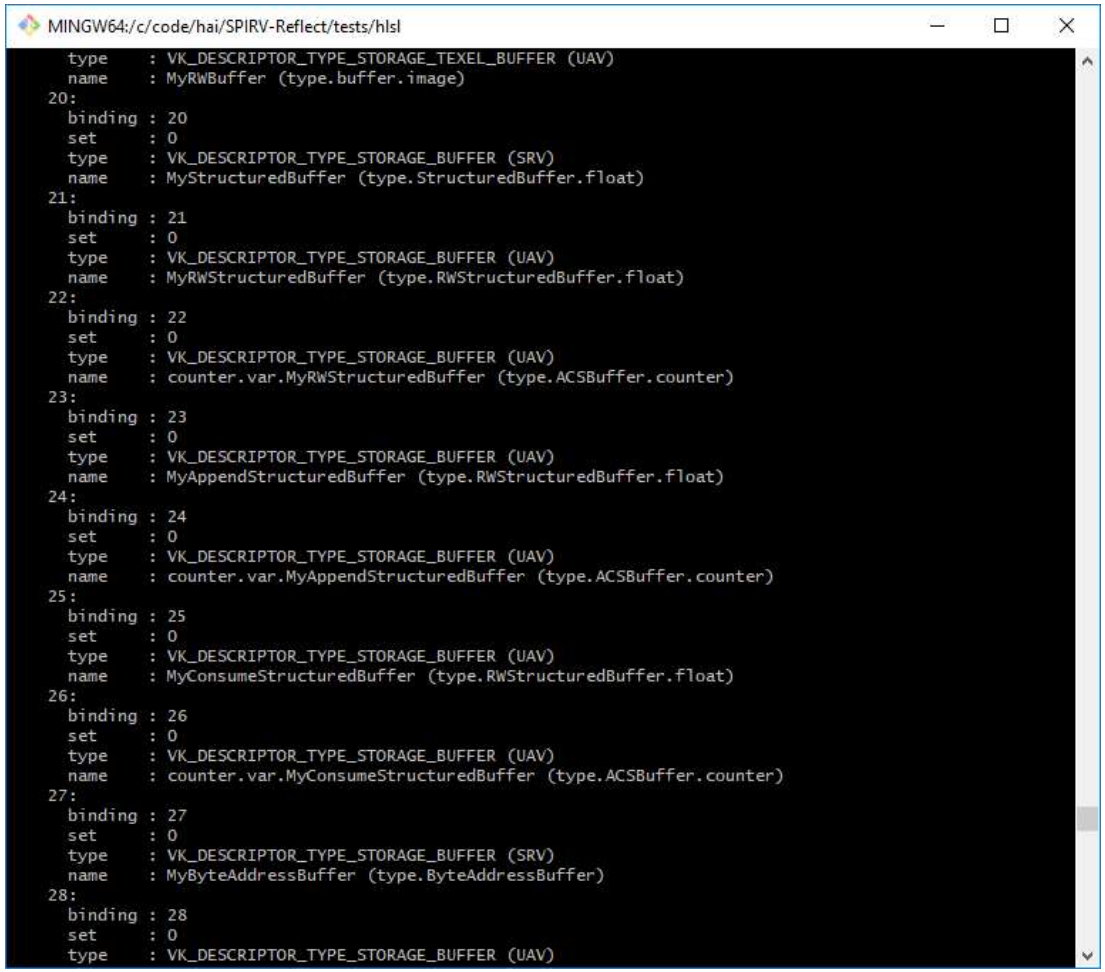
- **SPIRV-Reflect**
  - Small library C/C++
    - 2 files: 1 header, 1 source
  - Cort Stratton + Hai Nguyen
  - Send bugs and requests to Cort
- **Reflected Data**
  - Vertex attribute locations
    - Basic type info
  - Descriptor bindings and sets
    - Binding #, set#, descriptor type
  - Uniform, storage, push constants blocks
    - Relative offsets, absolute offsets, raw size, padded size, type info
  - HLSL resource types
    - CBV, SRV, UAV, Samplers
  - Semantics and Counter Buffers

I <3 BUGS!

What's Here: Reflection



What's Here: Reflection



What's Here: Reflection

- https://github.com/chaoticbob/SPIRV-Reflect
- **Goes live this Friday (March 23, 2018)**
- **Please file issues on Github**
- **Work in progress to add reflection to SPIRV-Tools**
  - Will deprecate SPIRV-Reflect

I <3 ISSUES!

What's Coming

- **Descriptor Indexing!**
  - NonUniformResourceIndex FTW!
  - "Coming soon"
- **Extensions Support**
  - Will likely be command line option
    - -fspv-extension=<ext-a> -fspv-extension=<ext-b>
- **Moar SM 6.x**
  - 64-bit integer
  - 16-bit scalars
  - Barycentrics
- **Things being considered…but no firm conclusion**
  - Inline cbuffer initialization
  - Root descriptors in HLSL source
  - Specifying extensions in the source like GLSL

Credits

- **Individuals** (apologies if I missed anyone)
  - **Team #spiregg (DXC/SPIRV)**
    - Lei Zhang (Google), Ehsan Nasiri (Google)
  - **Guidance**
    - David Neto (Google), John Kessenich (Google)
  - **spirv-opt**
    - Greg Fischer(LunarG), Diego Novillio (Google), Steven Perron (Google), Alan Baker (Google)
  - **IHV Friends**
    - Dr. Matthäus G. Chajdas (AMD), Nuno Subtil (NVIDIA), Piers Daniell (NVIDIA), Jason Ekstrand (Intel), Slawomir Grajewski (Intel)
  - **Khronos Members**
    - Neil Henning (Codeplay), Tobias Hector (Imagination), Dan Ginsburg (Valve)
  - **Community Members**
    - Graham Wihlidal (EA), Andrew Lauritzen (EA), Dan Baker (Oxide)
- **Companies**
  - AMD, Intel, LunarG, Microsoft, Nvidia, Valve

Thank You!