


UNREAL ENGINE

EFFICIENT USE OF VULKAN

Niklas "Smedis" Smedberg

Technical Director, Platform Partnerships, Epic Games

SIGGRAPH 2016

UNREAL ENGINE 

UE4 DEMO: PROTOSTAR

Collaboration between Epic Games and Samsung

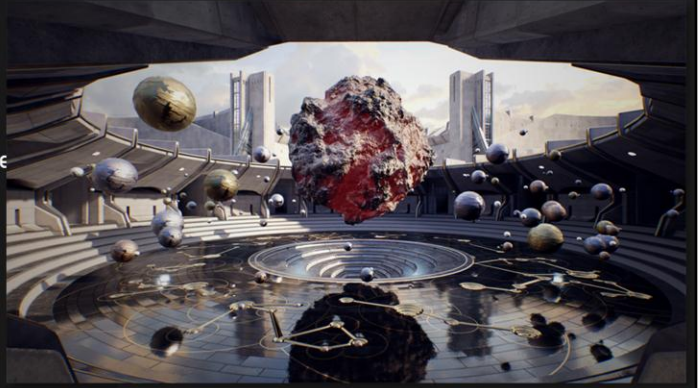


Goals:

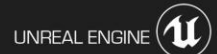
- Impressive Vulkan graphics
- Run on Samsung Galaxy S7
- Support Vulkan in Unreal Engine

Requirements:

- 1080p full HD
- 30 frames/sec



SIGGRAPH 2016



UE4 PROTOSTAR



Showing UE4 ProtoStar video, as well as running in real-time on a Samsung Galaxy S7 device.

UE4 PROTOSTAR - FEATURES

New support for console-style graphics effects on mobile:

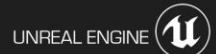


Depth of Field



GPU Particles

SIGGRAPH 2016



Examples of graphics features that we've previously implemented on dedicated gaming consoles, but never on mobile.

UE4 PROTOSTAR - ONE MORE THING

Real-time planar reflections for dynamic objects:



SIGGRAPH 2016

UNREAL ENGINE 

A completely new graphics feature that we've never had before, on any platform.

GPU TILE MEMORY

Mobile GPUs use a tile-based hardware architecture

Screen is split into tiles (e.g. 32x32 pixels)

The whole tile fits inside the GPU, on-chip

Extremely fast access

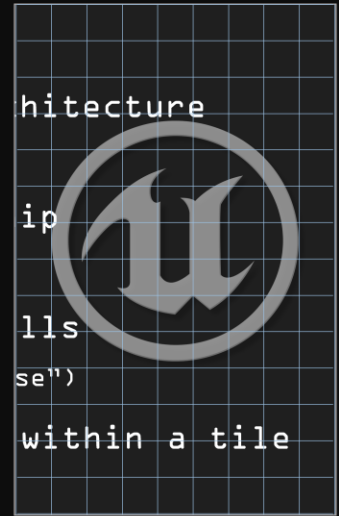
Process all vertex shaders for all draw calls

Sort screen-space triangles per tile ("binning phase")

Process all pixel shaders for all pixels within a tile

When finished, save final tile memory to RAM

Repeat for all tiles, until the whole screen has



ADVANTAGES OF TILE MEMORY

On-chip tile memory is extremely fast

Some tile-based GPUs have dedicated tile memory in each GPU "core"

- GPU cores can then process multiple tiles in parallel
- Easy to scale hardware performance

Bulk data transfer between tile memory and RAM

- Loss-less bandwidth compression
- Can do MSAA-downsampling at the same time (less data to transfer, "free" MSAA resolve)
- Can clear your rendertarget for free

VULKAN - RENDER PASSES

OpenGL never understood what "tile memory" is:

- Changing framebuffer may or may not do Bad Things

- Driver has to guess (often based on glClear)

- May unnecessarily load old framebuffer from RAM (slow)

- May unnecessarily store all tile memory to RAM (slow)

But Vulkan knows this, and specifies each render pass explicitly:

```
vkBeginRenderPass()
```

```
vkEndRenderPass()
```

Explicitly specified load/store operations!

FRAMEBUFFER FETCH

What if your pixelshader could have direct access to the rendertarget data?

- Custom alpha-blend operations

- Fade out smoke particles to avoid hard intersections with the game environment

- Deferred rendering with G-buffer

With GPU tile memory, you can do this for free!

OpenGL ES extension:

- GL_EXT_shader_framebuffer_fetch

- Access background color in the pixelshader as "gl_LastFragData"

PIXEL LOCAL STORAGE

A more powerful OpenGL ES extension:

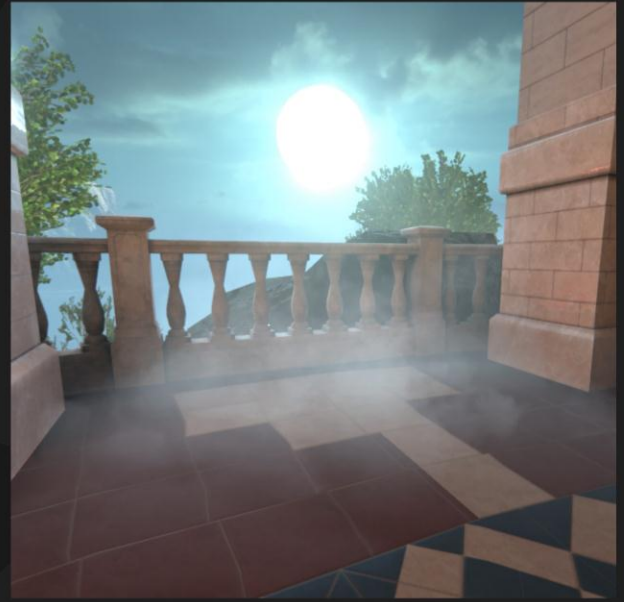
```
EXT_shader_pixel_local_storage
```

Allows access to full MRT data, not just one background color

Exposes tile memory as a struct in the pixelshader:

```
__pixel_localEXT FragDataLocal  
{  
    layout(r32f) highp float_value;  
    layout(r11f_g11f_b10f) mediump vec3 normal;  
    layout(rgb10_a2) highp vec4 color;  
    layout(rgba8ui) mediump uvec4 flags;  
} pls;
```

CUSTOM DEPTH-FADE USING TILE MEMORY



VULKAN "MULTIPASS"

"Multipass" allows staying on GPU for multiple render passes!

Vulkan allows the driver to "fuse" together subsequent sub-passes, automatically

API to specify input and output buffers for each pass

Allows mobile tile-based GPUs to stay within a single tile during multiple passes

Measured to be about 30% faster when using Multipass

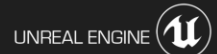
Similar to "framebuffer fetch" and "pixel local storage":

But works cross-platform, with no extensions needed!

The API is all on the C++ side, no shader source code changes required

Great for tile-based GPUs, and no downside for other GPUs

SIGGRAPH 2016



Fusing of sub-passes is done in particular if there are shared input/output buffers between sub-passes.

More but simpler shaders (or passes) can be faster than fewer but more complex shaders, because simpler shaders may require fewer number of temp registers.

Less temp register usage improves GPU utilization (more GPU threads in parallel, better latency-hiding).

VULKAN PIPELINE STATE

OpenGL:

- Set vertex and pixel shaders
- Set blending states
- Set depth states
- Set rasterizer states
- Etc...

Vulkan puts all important states in a single object

- Very efficient to switch all states in one go!
- ...But it's slow to create a pipeline state object

PIPELINE STATE CACHE

Caching compiled OpenGL shaders:

Driver may try to cache compiled shaders

Hitches and slow performance on first run, but fast on subsequent runs

Extension to save "program binaries"

Or try to make dummy drawcalls during load time

Caching compiled Vulkan shaders in UE4:

Pipeline states (and shaders) can be saved to a cache file

Save the cache file and use during load time for "no hitches on first run"!

We wanted ProtoStar to be fast even on the first run, not just subsequent runs.
That's why we implemented the pipeline state cache for Vulkan.

DOUBLE SPEED

FP16 shader instructions are twice as fast as fp32 instructions!

This is now spreading to other platforms than mobile!

The new NVIDIA Pascal GPU now supports FP16 double-speed elsewhere too

Wouldn't *you* like your next laptop or PC to suddenly be 2x faster overnight?

Let's get this on all platforms, not just mobile. Now.

MULTI-THREADED RENDERING

OpenGL and DirectX was never good for multi-threaded rendering.

Vulkan and DirectX 12 fix this:

Allows graphics programmers to fully utilize all CPU cores in parallel

Adding faster hardware CPU cores is expensive

But adding more hardware CPU cores is cheap

"Paragon" from Epic Games is an example of multi-threaded rendering in UE4.

VULKAN VALIDATION LAYERS

Vulkan removes all validation from the run-time to development time

- Fantastic for debugging and game development
- Fantastic for run-time performance, by removing all that overhead!

OpenGL pays for this overhead all the time! For all drawcalls!

Summary:

1. Stay within tile memory using Vulkan Multipass
2. Don't create pipeline states in-game
3. FP16 shader instructions are twice as fast as FP32 instructions!
4. Vulkan supports multithreaded rendering
5. Vulkan validation layers are great

THANK YOU!

QUESTIONS?

Download the full UE4 source code from
github.

Start experimenting with Vulkan today!

SIGGRAPH 2016

UNREAL ENGINE 

In particular, look at the Vulkan API implementation in our VulkanRHI source code.

MOBILE - THERMAL LIMITS

CPU and GPU will change speed many times during each frame!

The device is trying to avoid overheating

But makes it difficult to provide a smooth real-time graphics experience

During the intro/logo screen, ProtoStar limits itself to 1 FPS

Allows the device to cool down (within 6 seconds) between runs

BONUS SLIDE!