Excellent Vulkan Examples

From Sascha Willems

Lina Liu Nov. 20 2022

Vulkan C++ Examples and Demos

Vulkan- The new generation graphics and compute API from Khronos

Vulkan C++ Examples and Demos

Shaders

1. How Vulkan use Shaders?
    1. Vulkan comsumes shaders in an intermediate representation called SPIR-V
        1. What is intermediate representation?
            1. Is a bytecode format as opposed to human-readable syntax like GLSL and HLSL.
            2. The bytecode format is called SPIR-V
            3. The bytecode format is a format that can used to write graphics and compute shaders
        2. Why use the bytecode format called SPIR-V?
            1. The compilers written by GPU vendors to turn shader code into native code are significantly less complex.
            2. If you use human-readable syntax like GLSL, some GPU vendors may rejecting your code due to syntax errors, even may compiler bugs, use SPIR-V can avoided such errors.

Vulkan C++ Examples and Demos

Shaders

1. How Vulkan use Shaders?
    3. How we write bytecode format shader?
        1. We don’t need to write the bytecode format by hand, Khronos released a compiler can compiles GLSL to SPIR-V.
        2. You can include this compiler as a library to produce SPIR-V at runtime.
        3. We can use compiler like glslangValidator.exe
        4. We can use compiler like glslc.exe
            1. What is good for using glslc?
                1. Glslc uses the same parameter format as well-known compilers like GCC and Clang and includes some extra functionality like includes.
        5. glslangValidator.exe and glslc.exe are included in Vulkan SDK.

Vulkan C++ Examples and Demos

A note on synchronization

1. Why is Synchronization important?
   1. Vulkan is explicitly parallel and built for multithreading.
   2. Vulkan can render scenes with maximum efficiency and minimal wait time.
   3. The key is making sure that any parallel tasks wait only when they need to, and only for as long as necessary.

Vulkan C++ Examples and Demos

A note on synchronization

2. How synchronization implemented in Vulkan?
    1. GPU queue
        1. Graphic operations
    2. CPU thread
        1. Command buffers
        2. Computing vertices
        3. Loading textures
    3. Cmd buffers from any CPU thread eventually inserted into the same GPU queue.
    4. The cmd in GPU queue can run in parallel, so no guarantee that the cmd will complete in the same order as in CPU thread.
    5. In-queue tools
        1. Pipeline barriers/ events/subpass dependencies

Vulkan C++ Examples and Demos

A note on synchronization

3. Synchronization at two levels
    1. Within a single queue
    2. Across multiple queues
        1. Semaphores
            1. Semaphores are for synchronizing solely between GPU tasks, especially across multiple queues, not for synchronizing between GPU and CPU tasks.
        2. Fences
            1. Fences are designed for GPU-to-CPU synchronization.

Vulkan C++ Examples and Demos

A note on Synchronization

4. A note on Synchronization
    1. Examples uses vkDeviceQueueWaitIdle at the end of each frame.
    2. Use vkDeviceQueueWaitIdle is a heavy operation and is suboptimal in regards to having CPU and GPU operations run in parallel.

Vulkan C++ Examples and Demos

Example lists

1. Basics
2. glTF
    1. These samples show how implement different features of the glTF 2.0 3D format transmission format in detail
3. Advanced
4. Performance
5. Physically Based Rendering
    1. A lightling technique that achieves a more realistic and dynamic look by applying approximations of bidirectional reluctance distribution functions based on measured real world material parameters and environment lighting.
6. Deferred
    1. Deferred examples use a deferred shading setup
7. Compute Shader
8. Geometry Shader
9. Tessellation Shader
10. Hardware accelerated ray tracing
11. Headless
    1. Examples that run one-time tasks and don’t make use of visual output(no window system integration). These can be run in environments where no user interface is available.
12. User interface
13. Effects
14. Extensions
15. Misc