The Vulkan Portability Enumeration Extension

Charles Giessen, LunarG, Inc.
April 2022

Introduction

Vulkan® Portability™ aims to counter platform fragmentation by encouraging layered implementations of Vulkan functionality over Metal, DX12, and other APIs. Vulkan Portability enables Vulkan applications to be reliably deployed across diverse platforms.

Khronos released a provisional version of Vulkan Portability Extension 1.0 in September 2020. The VK_KHR_portability_subset extension allows a non-conformant Vulkan implementation to be built on top of another non-Vulkan graphics API, and identifies the difference between that implementation and a fully-conformant native Vulkan implementation. The extension is detailed in the Vulkan Specification.

For already released applications that are expecting to see only fully Vulkan conformant devices, a backward-compatibility issue exists:

1. Applications already developed and released before the existence of the portability subset extension don't have a way to get only fully conformant physical devices when enumerating the devices available to an application. This could result in poor application behavior (performance, rendering artifacts, crashes, etc).
2. On Windows, Linux, and macOS, there is a mixture of conformant and non-conformant devices that exist today or in the future.
3. If a non-conformant device is used, validation of the VK_KHR_portability_subset for the non-conformant devices will generate validation errors if the application didn't query for the portability subset extension. But since the application was released before the creation of the portability subset extension, it doesn't know it should query the VK_KHR_portability_subset extension.

Thus, to maintain backward compatibility for applications that are expecting conformant devices, we need to ensure that the Vulkan loader only provides conformant physical devices from vkEnumeratePhysicalDevices().

Solving the Backward Compatibility Issue

To solve the backward compatibility issue, Khronos has released the new Vulkan Loader extension, VK_KHR_portability_enumeration. The purpose of this extension is to enable or disable enumeration of portability (non-conformant) implementations.

VK_KHR_portability_enumeration is a new Vulkan API extension for the portability initiative that allows applications to opt into enumerating portability devices. This means that applications not designed to work with the VK_KHR_portability_subset extension won't accidentally find physical devices which support the subset extension.

Usage is straightforward. First, because this is an instance extension, an application must check for and enable the extension in VkInstanceCreateInfo::ppEnabledLayerNames. The name of the extension is available through the macro
VK_KHR_PORTABILITY_ENUMERATION_EXTENSION_NAME.
Now add the VK_INSTANCE_CREATE_ENUMERATE_PORTABILITY_BIT_KHR flag to VkInstanceCreateInfo::flags and create an instance. Physical devices that support the VK_KHR_portability_subset extension will now be enumerated in vkEnumeratePhysicalDevices.

A possible implementation is as follows:

```
VkInstanceCreateInfo instance_create_info{};
instance_create_info.flags =
   VK_INSTANCE_CREATE_ENUMERATE_PORTABILITY_BIT_KHR;
instance_create_info.enabledLayerCount = 1;
instance_create_info.ppEnabledLayerNames =
   { VK_KHR_PORTABILITY_ENUMERATION_EXTENSION_NAME };
...

VkInstance instance;
vkCreateInstance(&instance_create_info, NULL, &instance);

...
// Can now enumerate VkPhysicalDevices which support the
VK_KHR_porability_subset
vkEnumeratePhysicalDevices(instance, &count, ...);
```

If the application sees physical devices that support the VK_KHR_portability_subset extension, the application can now continue to query the portability subset and continue it's implementation.

Potential impact to existing MoltenVK applications

To realize the benefits of the new portability enumeration extension, applications will need to update the loader version in their application bundle. Applications that currently target the portability subset will need to enable the VK_KHR_portability_enumeration extension and modify their `VkInstanceCreateInfo` accordingly. This is because physical devices which support VK_KHR_portability_subset will no longer be provided by default without enabling the portability enumeration extension.

Phased release plan

The initial release of the portability enumeration extension (in the Vulkan 1.3.211.0 SDK) will not prevent applications from using portability devices and instead will issue a warning if the extension is not enabled. That way applications which don't enable the extension aren't suddenly unable to find portability devices after updating their loader. A subsequent release will then mandate the use of the portability enumeration extension.

Portability Drivers interface with Vulkan Loader

The way to create a 'portability driver' is to add the "is_portability_driver": true to the ICD Manifest json file. For more information on the interface between the driver and the Vulkan Loader, refer to the Loader-Driver-Interface documentation.