Enhanced Debugging with the Vulkan Loader

Mark Young
LunarG, Inc.

Presented at the Khronos Vulkanise 2023 Conference
Agenda

- Recap of Vulkan Desktop Loader and Layers
- Logging Improvements
- Filtering Environment Variables
- New Docs
Vulkan Desktop Loader Overview

Summary: Loader finds layers and drivers on your system
Vulkan Layer Types

● Implicit
  ○ Automatically loaded
    ■ Unless defines “enable” environment variable
  ○ Must define a “disable” environment variable
    ■ Loader detects “disable” then does not load
  ○ Typically loaded before explicit layers
  ○ Example:
    ■ Steam Fossilize

● Explicit
  ○ Selected by application, tools, or command-line environment variables
  ○ Example:
    ■ Validation
Vulkan Layer Intercepted Call-Chain

Example Instance Call-Chain

Example Device Call-Chain

Note:
If you query your own entrypoints with vkGetDeviceProcAddr, most device commands won’t include “Loader Trampoline” in call-chain.
Loader Logging Improvements

● Why focus on the Loader?
  ○ Everyone has loader (requires no extra tools)

● Goals of Improved Logging:
  ○ Easier diagnosis of warnings and errors
  ○ Improved analysis of layer and driver issues
  ○ Understand more about the environment discovered by the loader
Loader Debug Environment Variable

- **VK_LOADER_DEBUG**
  - Comma-delimited list of message levels of interest:
    - error, warn, info, debug, all

- Starting with Vulkan loader 1.3.205*, new options:
  - layer
  - driver

- Examples:
  - Windows:       set VK_LOADER_DEBUG=error,warn,layer
  - Linux/Mac:     export VK_LOADER_DEBUG=error,warn,driver

* 1.3.205 - GitHub: Feb 2022, Vulkan SDK: April 2022
Layer Discovery Logging

- Searched for during all pre-Instance and CreateInstance calls
- Separate searches for Implicit and Explicit Layers separately

LAYER: Searching for layer manifest files
LAYER: In following folders:
LAYER: /home/$USER/.config/vulkan/implicit_layer.d
LAYER: /etc/xdg/vulkan/implicit_layer.d
LAYER: /etc/vulkan/implicit_layer.d
LAYER: /home/$USER/.local/share/vulkan/implicit_layer.d
LAYER: /home/$USER/.local/share/flatpak/exports/share/vulkan/implicit_layer.d
LAYER: /var/lib/flatpak/exports/share/vulkan/implicit_layer.d
LAYER: /usr/local/share/vulkan/implicit_layer.d
LAYER: /usr/share/vulkan/implicit_layer.d
LAYER: Found the following files:
LAYER: /etc/vulkan/implicit_layer.d/renderdoc_capture.json
LAYER: /home/$USER/.local/share/vulkan/implicit_layer.d/steamfossilize_i386.json
LAYER: /home/$USER/.local/share/vulkan/implicit_layer.d/steamfossilize_x86_64.json
LAYER: /home/$USER/.local/share/vulkan/implicit_layer.d/steamoverlay_i386.json
LAYER: /home/$USER/.local/share/vulkan/implicit_layer.d/steamoverlay_x86_64.json
LAYER: /usr/share/vulkan/implicit_layer.d/nvidia_layers.json
LAYER: /usr/share/vulkan/implicit_layer.d/VkLayer_MESA_device_select.json
Driver Discovery Logging

- Searched for during all pre-Instance and CreateInstance calls

```
DRIVER: Searching for driver manifest files
DRIVER: In following folders:
DRIVER: /home/$USER/.config/vulkan/icd.d
DRIVER: /etc/xdg/vulkan/icd.d
DRIVER: /etc/vulkan/icd.d
DRIVER: /home/$USER/.local/share/vulkan/icd.d
DRIVER: /home/$USER/.local/share/flatpak EXPORTS/share/vulkan/icd.d
DRIVER: /var/lib/flatpak EXPORTS/share/vulkan/icd.d
DRIVER: /usr/local/share/vulkan/icd.d
DRIVER: /usr/share/vulkan/icd.d
DRIVER: Found the following files:
DRIVER: /usr/share/vulkan/icd.d/intel_icd.x86_64.json
DRIVER: /usr/share/vulkan/icd.d/lvp_icd.x86_64.json
DRIVER: /usr/share/vulkan/icd.d/radeon_icd.x86_64.json
DRIVER: /usr/share/vulkan/icd.d/lvp_icd.i686.json
DRIVER: /usr/share/vulkan/icd.d/radeon_icd.i686.json
DRIVER: /usr/share/vulkan/icd.d/intel_icd.i686.json
DRIVER: /usr/share/vulkan/icd.d/nvidia_icd.json
```
Loader vkCreateInstance Call-Chain Logging

- With “layer” enabled in VK_LOADER_DEBUG, loader will generate a rough instance call-chain during vkCreateInstance
  - Lists enabled implicit and explicit layers
  - If layer is implicit, it also details what its disable environment variable is

```
LAYER:  vkCreateInstance layer callstack setup to:
LAYER:    <Application>
LAYER:       ||
LAYER:     <Loader>
LAYER:       ||
LAYER:     VK_LAYER_MESA_device_select
LAYER:         Type: Implicit
LAYER:         Disable Env Var: NODEVICE_SELECT
LAYER:         Manifest: /usr/share/vulkan/implicit_layer.d/VkLayer_MESA_device_select.json
LAYER:         Library:  libVkLayer_MESA_device_select.so
LAYER:       ||
LAYER:     VK_LAYER_KHRONOS_validation
LAYER:         Type: Explicit
LAYER:         Manifest: /usr/share/vulkan/explicit_layer.d/VkLayer_khronos_validation.json
LAYER:         Library:  libVkLayer_khronos_validation.so
LAYER:       ||
LAYER:     <Drivers>
```
Loader vkCreateDevice Call-Chain Logging

- With “layer” and “driver” enabled in VK_LOADER_DEBUG, loader will generate a rough device call-chain during vkCreateDevice
  - Lists enabled implicit and explicit layers
  - Lists driver enabled by name and selected device info

INFO | LAYER: Failed to find vkGetDeviceProcAddr in layer libVkLayer_MESA_device_select.so
DRIVER | LAYER: vkCreateDevice layer callstack setup to:
DRIVER | LAYER: <Application>
DRIVER | LAYER: ||
DRIVER | LAYER: <Loader>
DRIVER | LAYER: ||
LAYER: VK_LAYER_KHRONOS_validation
LAYER: Type: Explicit
LAYER: Manifest: /usr/share/vulkan/explicit_layer.d/VkLayer_khronos_validation.json
LAYER: Library: libVkLayer_khronos_validation.so
LAYER: ||
DRIVER | LAYER: <Device>
DRIVER | LAYER: Using "NVIDIA GeForce GTX 1650" with driver: "libGLX_nvidia.so.0"

(Notice no VK_LAYER_MESA_device_select this time)
Loader Filter Environment Variables

● Previously
  ○ No way to disable layers or drivers easily
  ○ Enable required full layer name or driver manifest file
    ■ VK_INSTANCE_LAYERS
    ■ VK_DRIVER_FILES/VK_ICD_FILENAMES

● Starting in Vulkan Desktop Loader 1.3.234*

● Meant for Debugging

● CI systems could force specific layers and/or individual drivers per test scenario

* 1.3.234 - GitHub: Nov 2022, Vulkan SDK: Dec 2022
Filter Environment Variable Format

- Case insensitive
- Comma-delimited

Simple Globs
- Prefix: VKLayer*
- Suffix: *validation
- Substring: *KHRONOS*
- Whole name: VkLayer_Khronos_validation

Disable env var evaluated first, then enable
- Disable everything, the re-enable only what you want
Loader Layer Filter Environment Variables

- Enable/Disable Filter Environment Variables
  - VK_LOADER_LAYERS_ENABLE
  - VK_LOADER_LAYERS_DISABLE

- Special Layer Disable Globs
  - ~implicit~
  - ~explicit~
  - ~all~ or *

- Why Debug Only?
  - Disabling a layer that an application is relying on could have consequences
Example Layer Filter Environment Variables

- Disable all implicit layers
  - set VK_LOADER_LAYERS_DISABLE=~implicit~

- Disable all layers
  - set VK_LOADER_LAYERS_DISABLE=*  

- Disable all implicit layers, **except** if Valve is in name:
  - set VK_LOADER_LAYERS_DISABLE=~implicit~
  - set VK_LOADER_LAYERS_ENABLE=*valve*
Loader Driver Filter Environment Variables

- Select/Disable Filter Environment Variables
  - VK_LOADER_DRIVERS_SELECT
    - “Select” because all drivers enabled by default
  - VK_LOADER_DRIVERS_DISABLE
    - Names matched against driver manifest file name
      - For example: intel_icd.x86_64.json
  - Example:
    - Disable all drivers, except if Nvidia is in name:
      - set VK_LOADER_DRIVERS_DISABLE=*nvidia*
Investigating Bad Layer

```bash
$ vkcube
Selected GPU 1: NVIDIA GeForce GTX 1650, type: DiscreteGpu
Segmentation fault (core dumped)
```
Investigating Bad Layer (Debug messages)

```
$ VK_LOADER_DEBUG=layer vkcube
LAYER: searching for layer manifest files
LAYER: In following folders:
LAYER:  /home/marky/.config/vulkan/implicit_layer.d
LAYER:  /etc/xdg/vulkan/implicit_layer.d
LAYER:  /etc/vulkan/implicit_layer.d
LAYER:  /home/marky/.local/share/vulkan/implicit_layer.d
LAYER:  /usr/local/share/vulkan/implicit_layer.d
LAYER:  $LUNARG_INSTALL_DIR/share/vulkan/implicit_layer.d
LAYER:  /usr/share/vulkan/implicit_layer.d

LAYER: vkCreateInstance layer callstack setup to:
LAYER: <Application>
LAYER: | |
LAYER: | |
LAYER: | | VK_LAYER_LUNARG_monitor
LAYER: Type: Implicit
LAYER: Disable Env Var: DISABLE_OVERRIDE_MONITOR
LAYER: Manifest: /home/marky/.local/share/vulkan/implicit_layer.d/VkLayer_override_monitor.json
LAYER: Library: /home/marky/.local/share/vulkan/implicit_layer.d/../libVkLayer_override_monitor.so
LAYER: | |
LAYER: | | VK_LAYER_MESA_device_select
LAYER: Type: Implicit
LAYER: Disable Env Var: NODEVICE_SELECT
LAYER: Manifest: /usr/share/vulkan/implicit_layer.d/VkLayer_MESA_device_select.json
LAYER: Library: libVkLayer_MESA_device_select.so
LAYER: | |
LAYER: | <Drivers>
```
Investigating Bad Layer (Disable All Layers)

```
$ VK_LOADER_DEBUG=layer VK_LOADER_LAYERS_DISABLE=* vkcub

Layer: Searching for layer manifest files
Layer: In following folders:
Layer: /home/marky/.config/vulkan/implicit_layer.d
Layer: /etc/xdg/vulkan/implicit_layer.d
Layer: /etc/vulkan/implicit_layer.d
Layer: /home/marky/.local/share/vulkan/implicit_layer.d
Layer: /home/marky/.local/share/flatpak/experts/share/vulkan/implicit_layer.d
```

Layer: vkCreateInstance layer callstack setup to:
Layer: <Application>
Layer:   |
Layer:   <Loader>
Layer:   |
Layer:   <Drivers>

Selected GPU 0: NVIDIA GeForce GTX 1650, type: DiscreteGpu
Layer: vkCreateDevice layer callstack setup to:

---

**LUNARG** 19
Investigating Bad Layer (Re-enable Device Select)

```
> $ VK_LOADER_DEBUG=layer VK_LOADER_LAYERS_DISABLE=* VK_LOADER_LAYERS_ENABLE=*mesa* vkcube
LAYER: Searching for layer manifest files
LAYER: In following folders:
LAYER: /home/marky/.config/vulkan/implicit_layer.d
LAYER: /etc/xdg/vulkan/implicit_layer.d
LAYER: /etc/vulkan/implicit_layer.d
LAYER: /home/marky/.local/share/vulkan/implicit_layer.d
LAYER: /home/marky/.local/share/flatpak/exports/share/vulkan/implicit_layer.d
LAYER: /usr/share/vulkan/implicit_layer.d
LAYER: /usr/share/flatpak/exports/share/vulkan/implicit_layer.d
LAYER: Loading layer library libVkLayer_MESA_device_select.so
INFO | LAYER: Insert instance layer "VK_LAYER_MESA_device_select" (libVkLayer_MESA_device_select.so)
LAYER: vkCreateInstance layer callstack setup to:
LAYER: 
LAYER: |<Application>
LAYER: |
LAYER: |<Loader>
LAYER: |
LAYER: |<Drivers>
LAYER: VK_LAYER_MESA_device_select
LAYER: Type: Implicit
LAYER: Disable Env Var: NODERVICE_SELECT
LAYER: Manifest: /usr/share/vulkan/implicit_layer.d/VkLayer_MESA_device_select.json
LAYER: Library: libVkLayer_MESA_device_select.so
```

Investigating Bad Layer (Look For Layer Disable)

```
$ VK_LOADER_DEBUG=layer vkcube
LAYER: Searching for layer manifest files
LAYER: In following folders:
LAYER: /home/marky/.config/vulkan/implicit_layer.d
LAYER: /etc/xdg/vulkan/implicit_layer.d
LAYER: /etc/vulkan/implicit_layer.d
LAYER: /home/marky/.local/share/vulkan/implicit_layer.d

LAYER: vkCreateInstance layer callstack setup to:
LAYER: <Application>
LAYER:   ||
LAYER:   <Loader>
LAYER:   ||
LAYER: VK_LAYER_LUNARG_monitor
LAYER:   Type: Implicit
LAYER:     Disable Env Var: DISABLE_OVERRIDE_MONITOR
LAYER:     Manifest: /home/marky/.local/share/vulkan/implicit_layer.d/VkLayer_override_monitor.json
LAYER:     Library: /home/marky/.local/share/vulkan/implicit_layer.d/../libVkLayer_override_monitor.so

LAYER: VK_LAYER_MESA_device_select
LAYER:   Type: Implicit
LAYER:     Disable Env Var: NODEVICE_SELECT
LAYER:     Manifest: /usr/share/vulkan/implicit_layer.d/VkLayer_MESA_device_select.json
LAYER:     Library: libVkLayer_MESA_device_select.so
LAYER:     ||
LAYER: <Drivers>
```
Investigating Bad Layer (Disable Only Bad Layer)

```bash
$ VK_LOADER_DEBUG=layer DISABLE_OVERRIDE_MONITOR=1 vkcube
```

Layer: Searching for layer manifest files
Layer: In following folders:
Layer: /home/marky/.config/vulkan/implicit_layer.d
Layer: /etc/xdg/vulkan/implicit_layer.d
Layer: /etc/vulkan/implicit_layer.d
Layer: /home/marky/ local/share/vulkan/implicit_layer.d

Debug | Layer: Loading layer library libVkLayer_MESA_device_select.so
Info | Layer: Insert instance layer "VK_LAYER_MESA_device_select" (libVkLayer_MESA_device_select.so)
Layer: vkCreateInstance layer callstack setup to:
Layer: <Application>
Layer:
Layer: <Loader>
Layer:
Layer: VK_LAYER_MESA_device_select
Layer: Type: Implicit
Layer: Disable Env Var: NODEVICE_SELECT
Layer: Manifest: /usr/share/vulkan/implicit_layer.d/VkLayer_MESA_device_select.json
Layer: Library: libVkLayer_MESA_device_select.so
Layer:
Layer: <Drivers>
New Docs

- Loader Debugging Markdown in Loader Repository
  - [https://github.com/KhronosGroup/Vulkan-Loader/blob/master/docs/LoaderDebugging.md](https://github.com/KhronosGroup/Vulkan-Loader/blob/master/docs/LoaderDebugging.md)

- “The Vulkan Loader and Vulkan Layers: Diagnosing Layer Issues” whitepaper
Shout Out!

● Charles Giessen
  ○ Current Vulkan Desktop Loader owner
  ○ Moderator on Vulkan Discord

● Community Involvement
  ○ Helps us continually improve the Desktop Loader
Questions?