Eurographics 2022

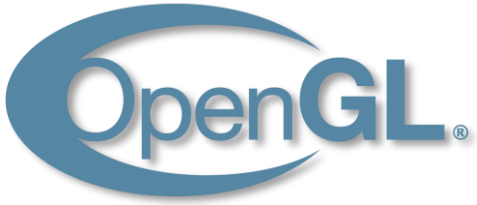
The Road to Vulkan

Teaching Modern Low-Level APIs in Introductory Graphics Courses

Johannes Unterguggenberger, Bernhard Kerbl,
and Michael Wimmer

TU Wien, Institute of Visual Computing &
Human-Centered Technology, Austria

Introductory Graphics Courses



1st contact with graphics APIs

2nd encounter with rasterization

~ 3rd semester, 3 ECTS

~ 150 students per year

Introductory Graphics Courses

1st contact with graphics APIs

2nd encounter with rasterization

~ 3rd semester, 3 ECTS

~ 150 students per year

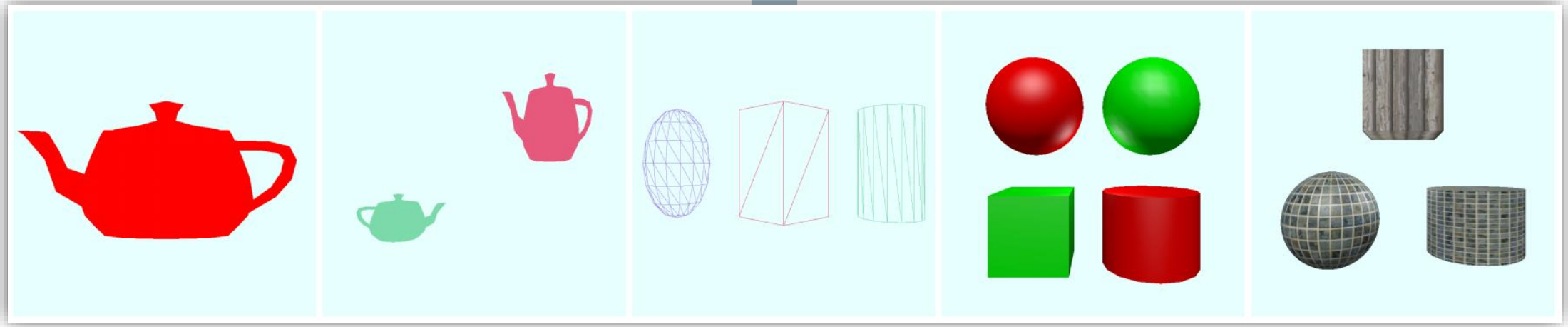
Assignment 1: Basic setup, window creation

Assignment 2: Transformations, camera/view

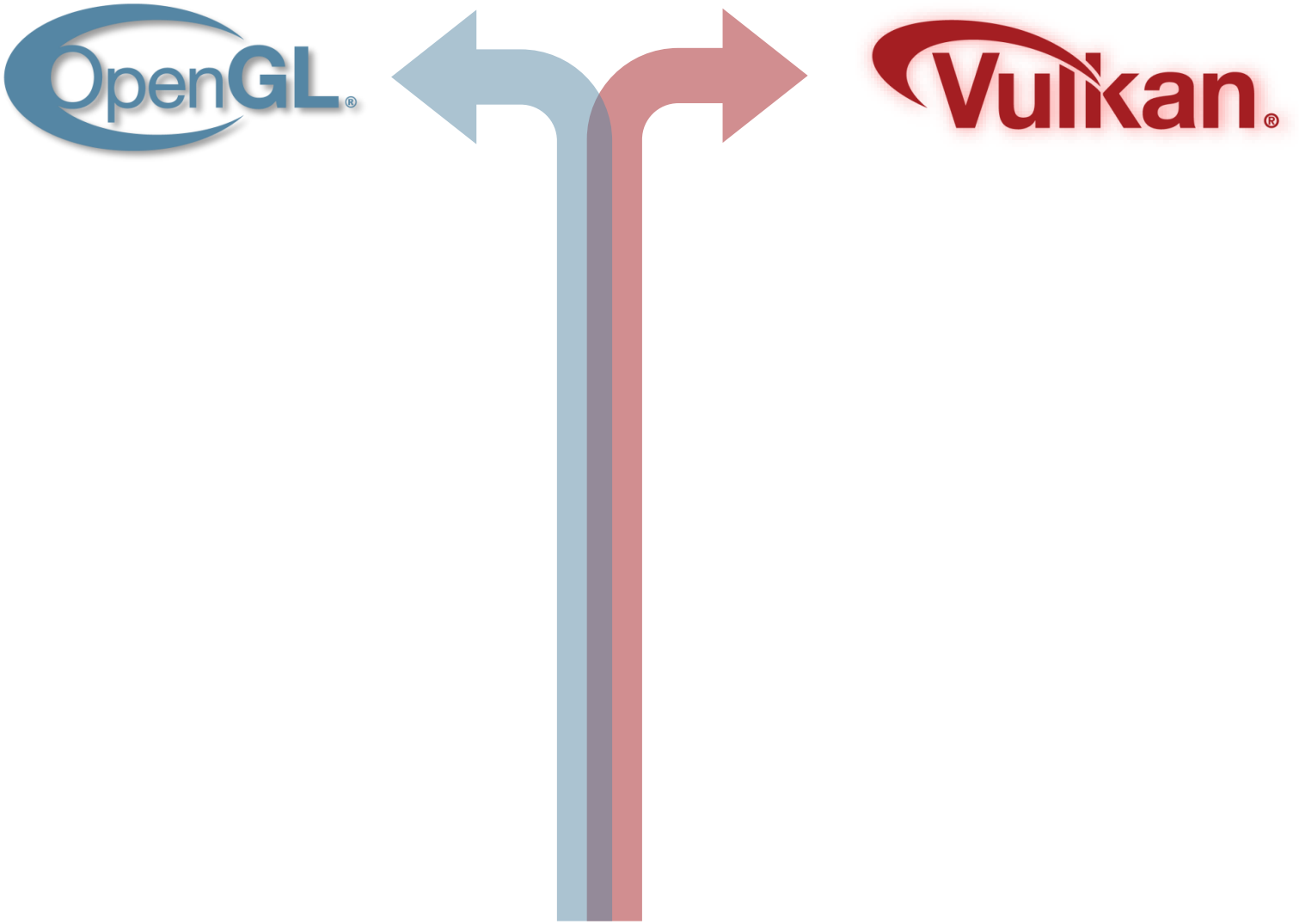
Assignment 3: Geometry, buffer handling

Assignment 4: Shader programming, lighting

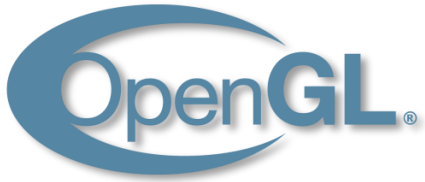
Assignment 5: Texturing



Introductory Graphics Courses

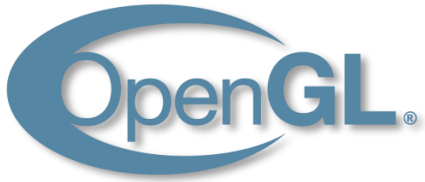


Introductory Graphics Courses



OpenGL		Vulkan
almost 30 years old	Age	just turned 6
high-level, complex drivers	Level	low-level, close to the metal
big huge state machine	API Design	parallelism and flexibility first
close to ancient hardware	Abstraction	close to modern hardware
concepts of the past	Insights	actual hardware operations
The Khronos Group, Inc.	Maintainer	The Khronos Group, Inc.

Introductory Graphics Courses

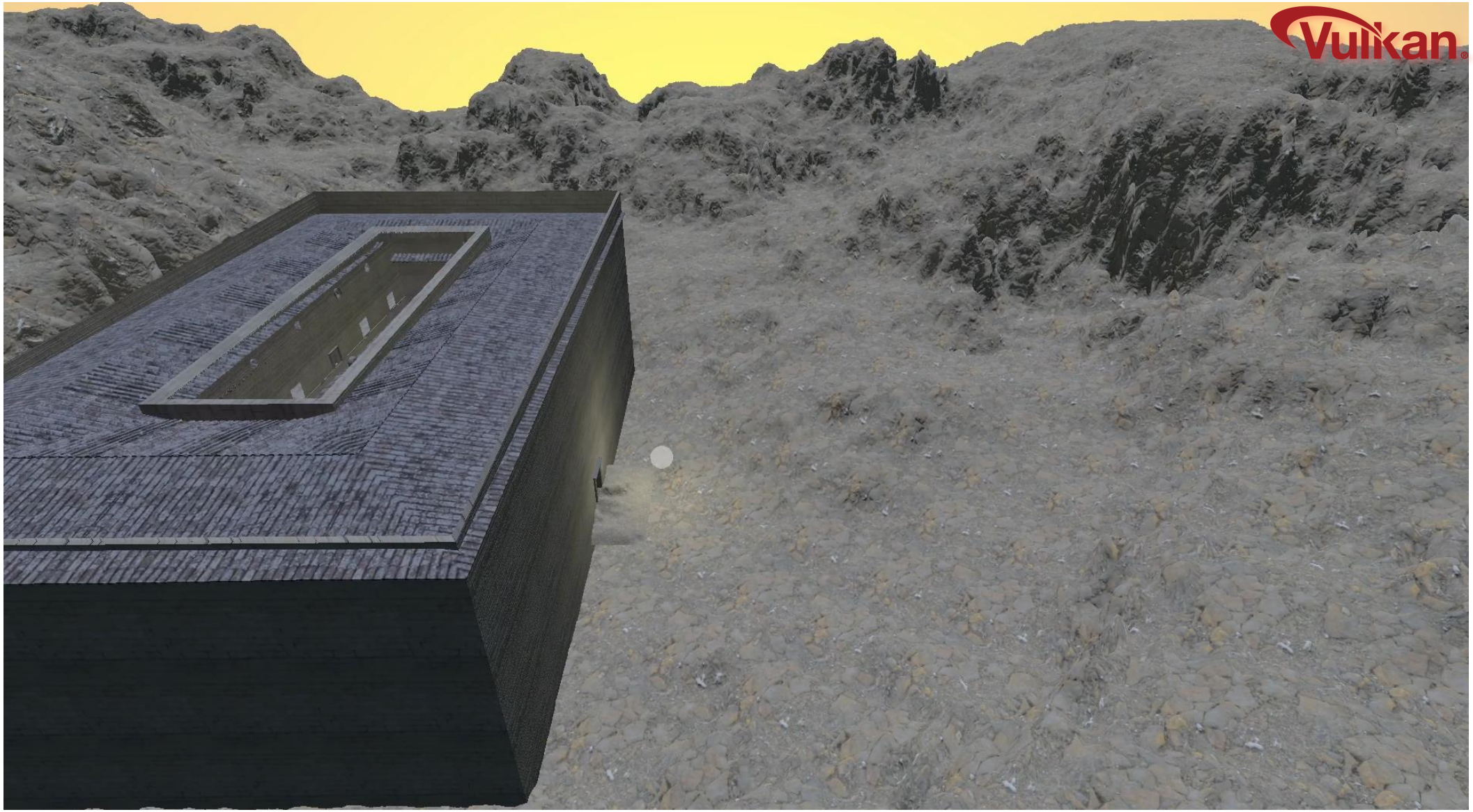


OpenGL		Vulkan
almost 30 years old	Age	just turned 6
high-level, complex drivers	Level	low-level, close to the metal
big huge state machine	API Design	parallelism and flexibility first
close to ancient hardware	Abstraction	close to modern hardware
concepts of the past	Insights	actual hardware operations
The Khronos Group, Inc.	Maintainer	The Khronos Group, Inc.

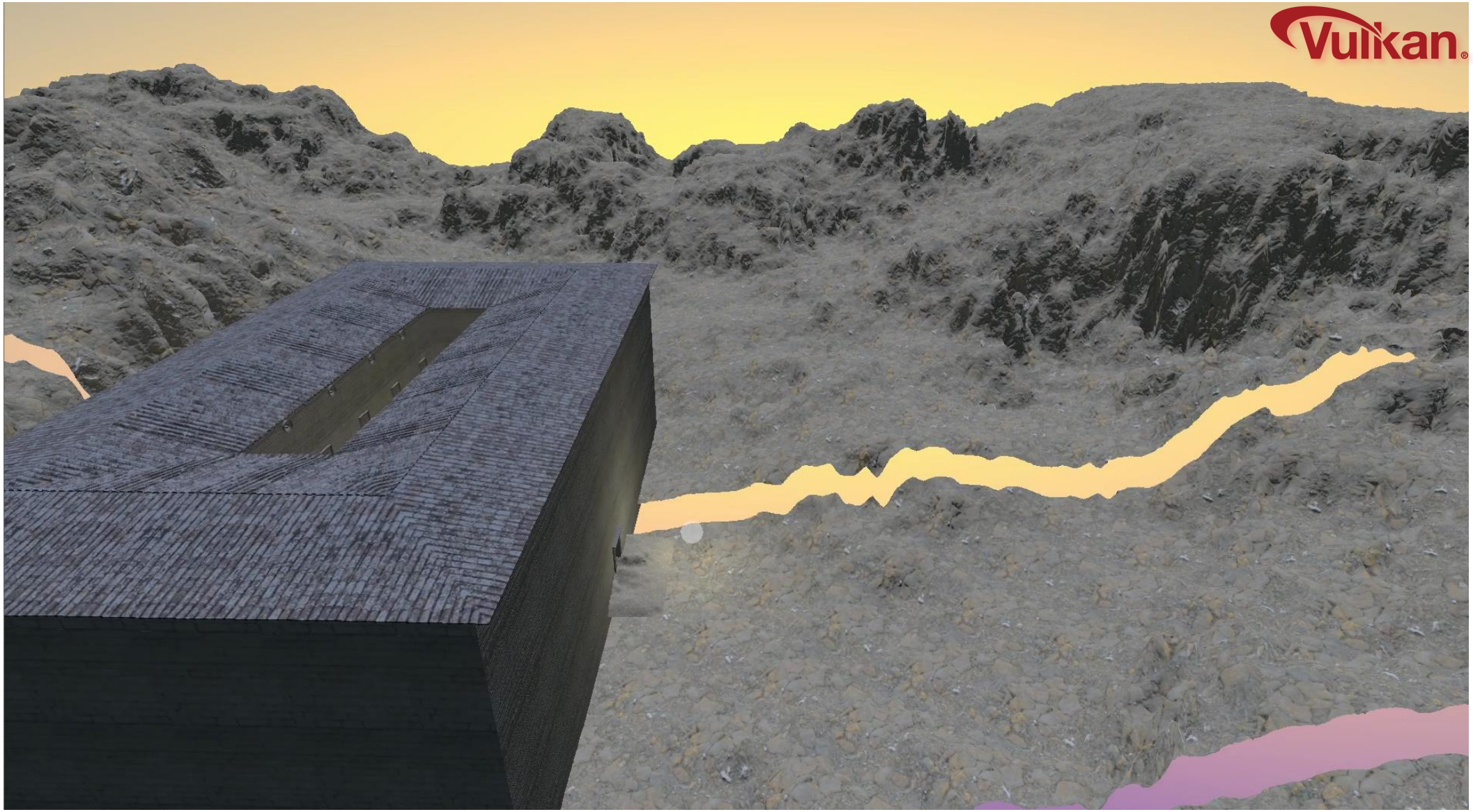
An Application Implemented in OpenGL

The Same Application Implemented in Vulkan

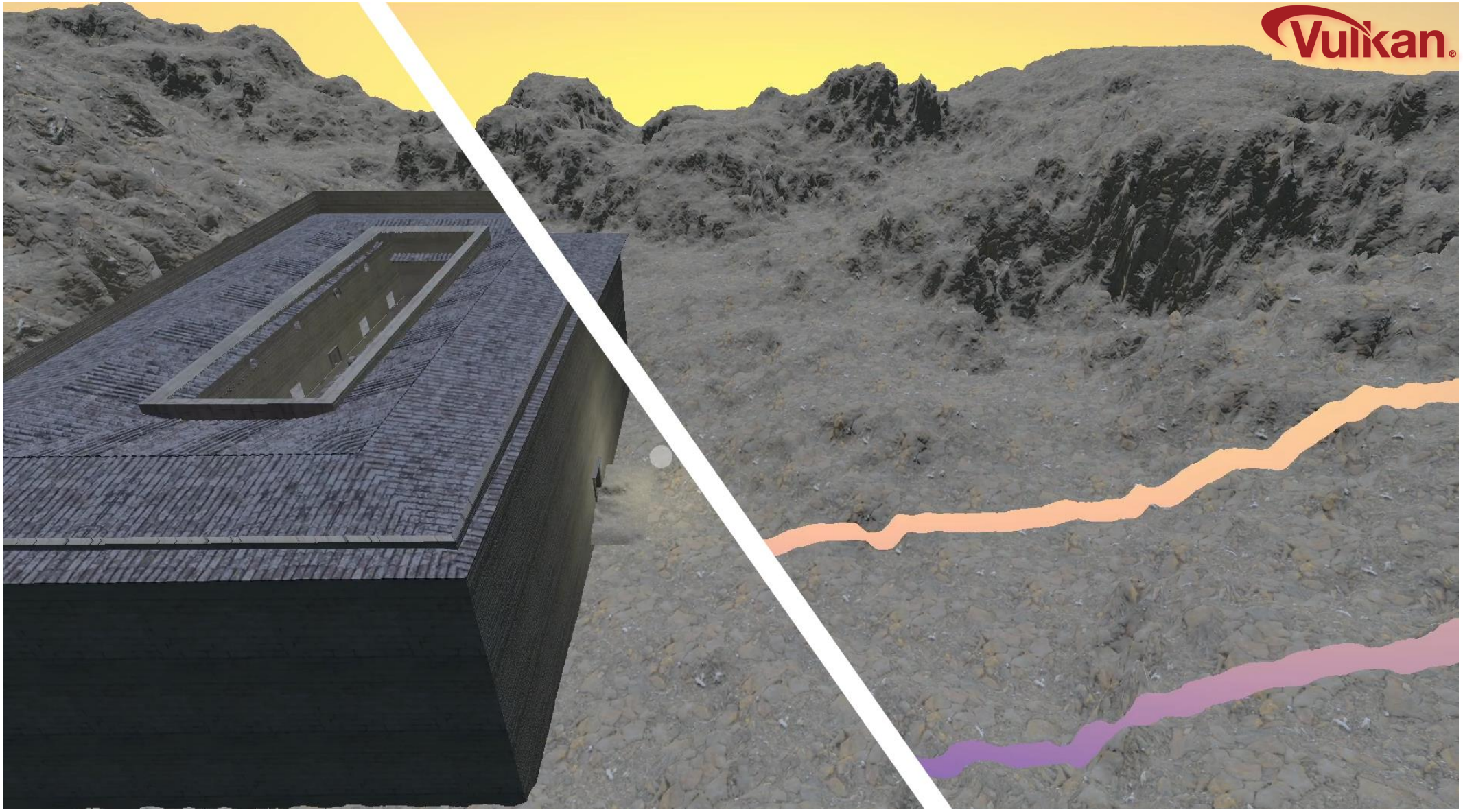
The Same Application Implemented in Vulkan



The Same Application Implemented in Vulkan



The Same Application Implemented in Vulkan



Vulkan Application Configuration

DEVICE/QUEUE

HOST

Draw Call 1

Vulkan Application Configuration

DEVICE/QUEUE

HOST

Draw Call 1

Draw Call 2

Vulkan Application Configuration

DEVICE/QUEUE

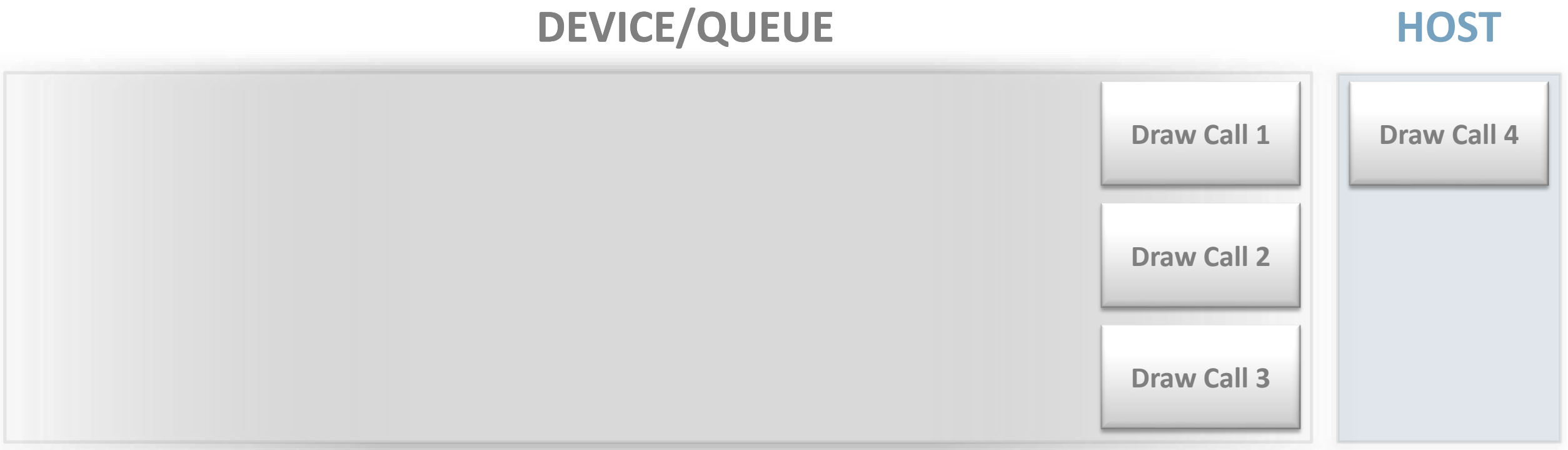
HOST

Draw Call 1

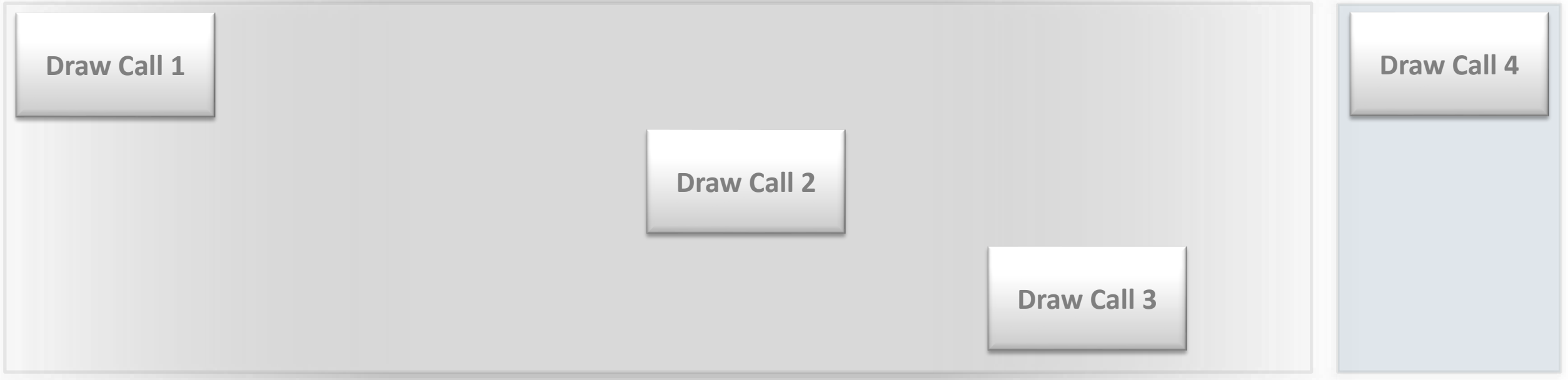
Draw Call 2

Draw Call 3

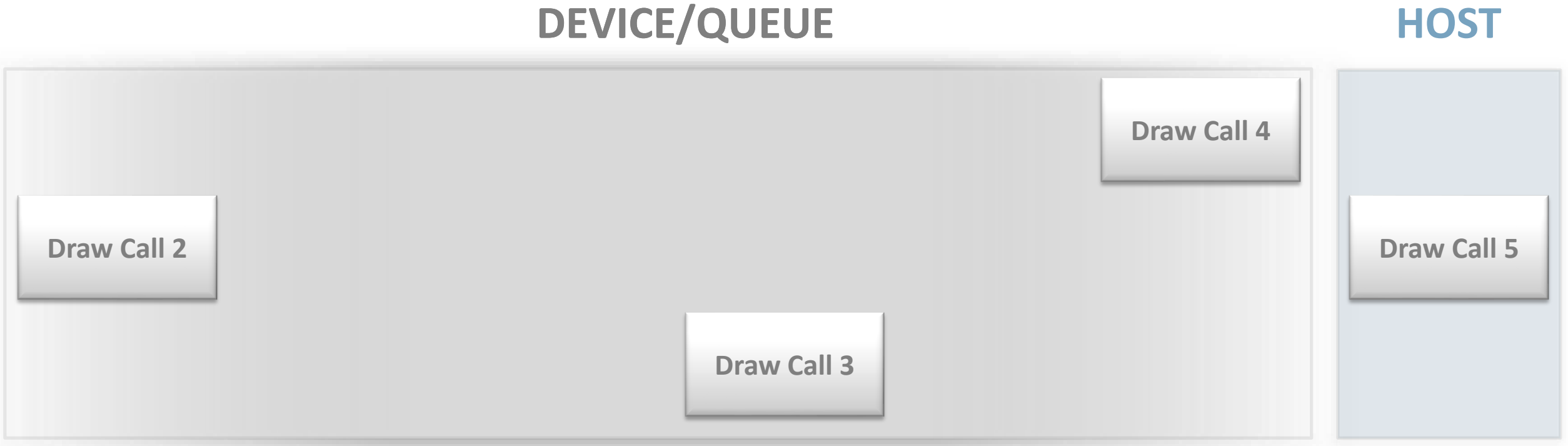
Vulkan Application Configuration



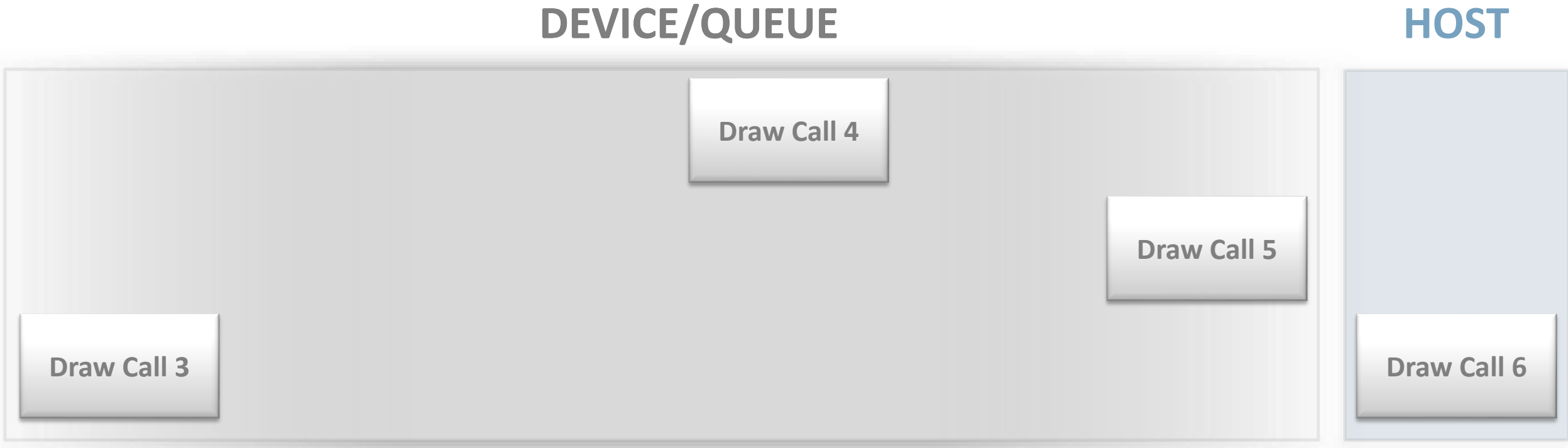
Vulkan Application Configuration



Vulkan Application Configuration



Vulkan Application Configuration



Vulkan Application Configuration

Uniform Buffer

DEVICE/QUEUE

HOST

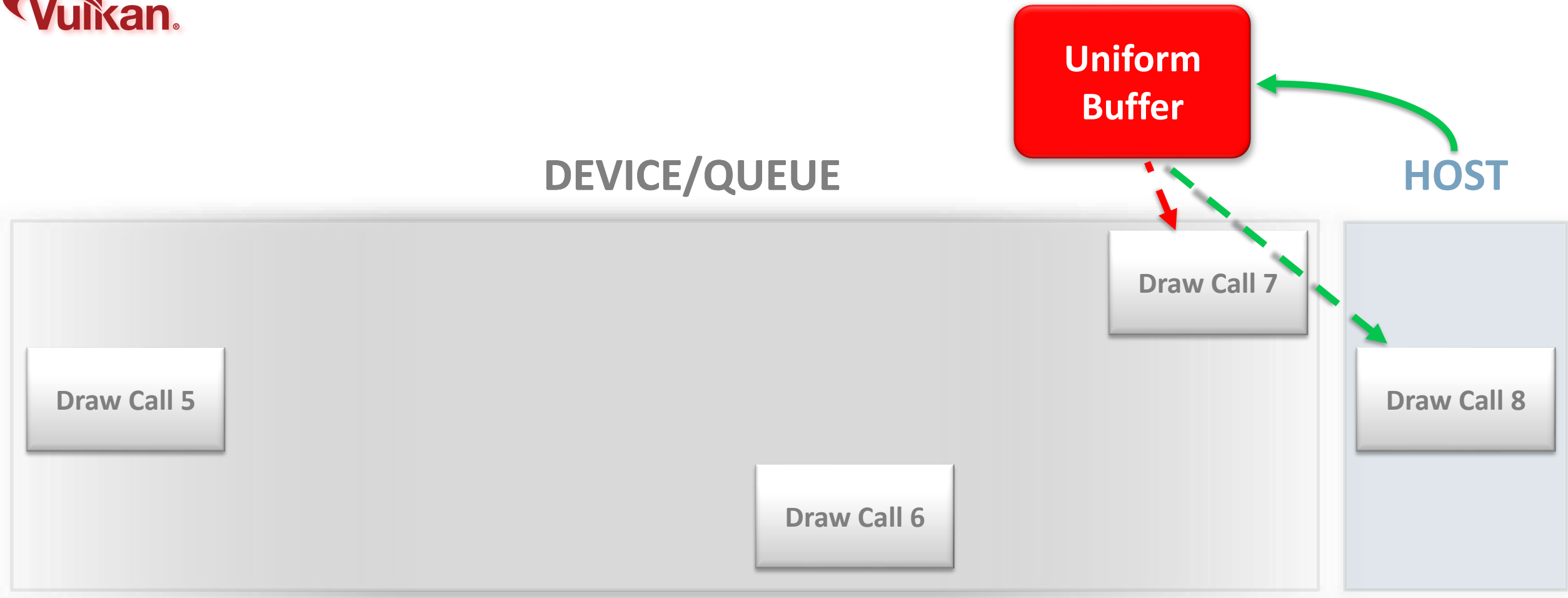
Draw Call 4

Draw Call 5

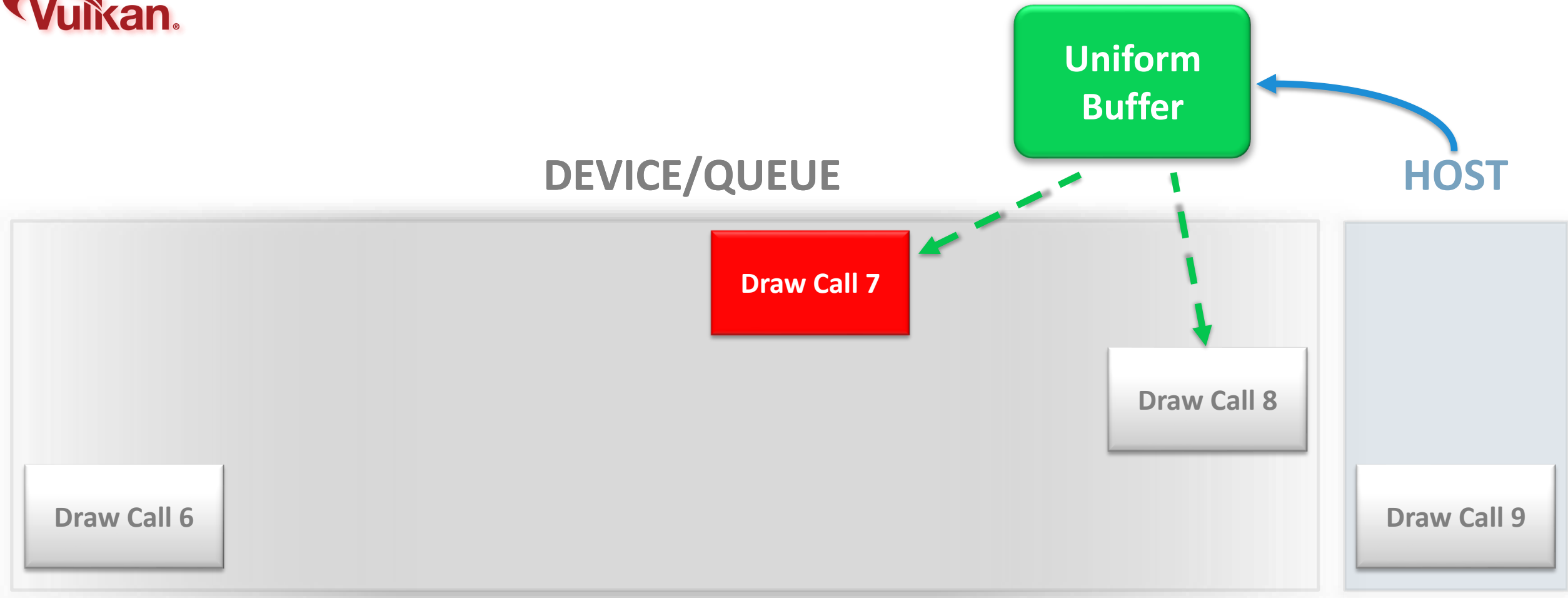
Draw Call 6

Draw Call 7

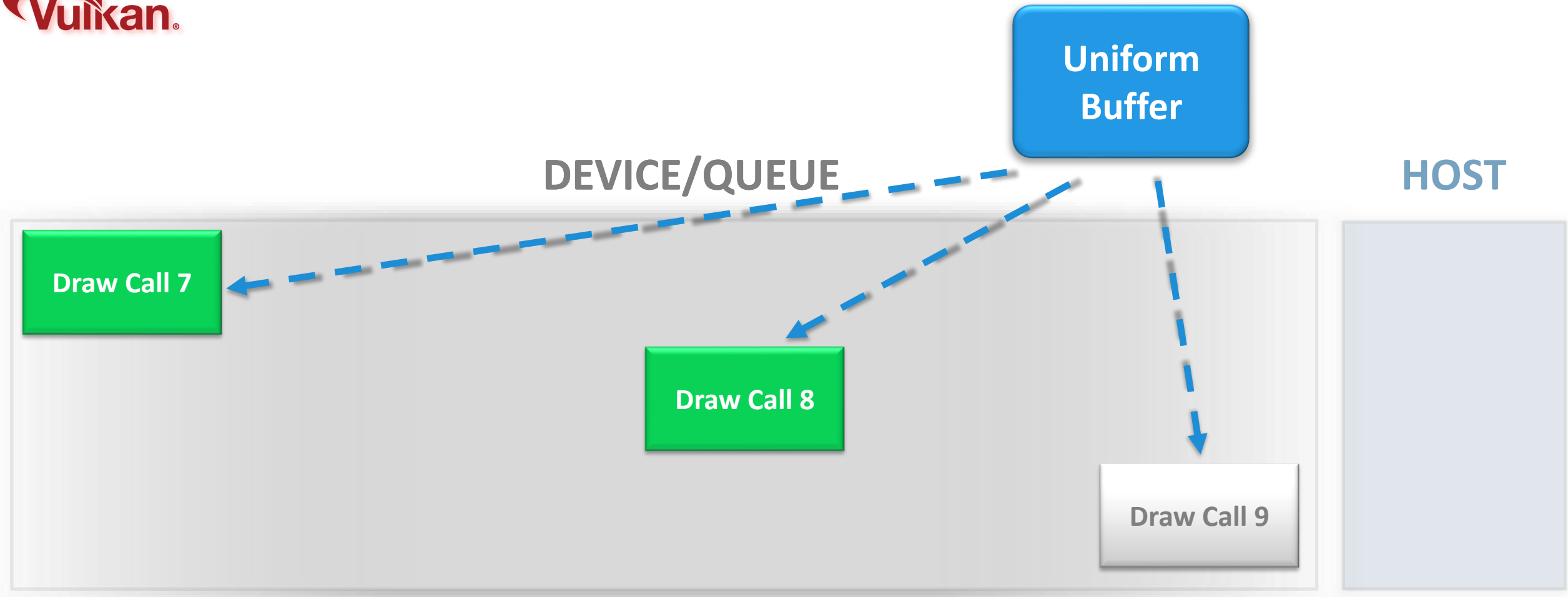
Vulkan Application Configuration



Vulkan Application Configuration



Vulkan Application Configuration



Vulkan Application Configuration

Uniform Buffer

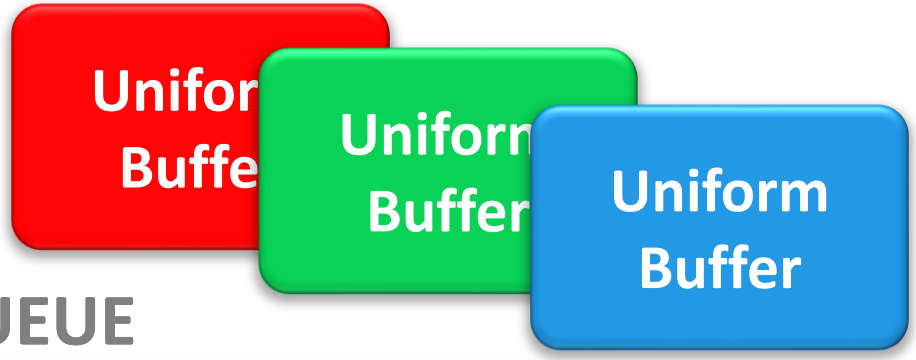
DEVICE/QUEUE

HOST

Draw Call 8

Draw Call 9

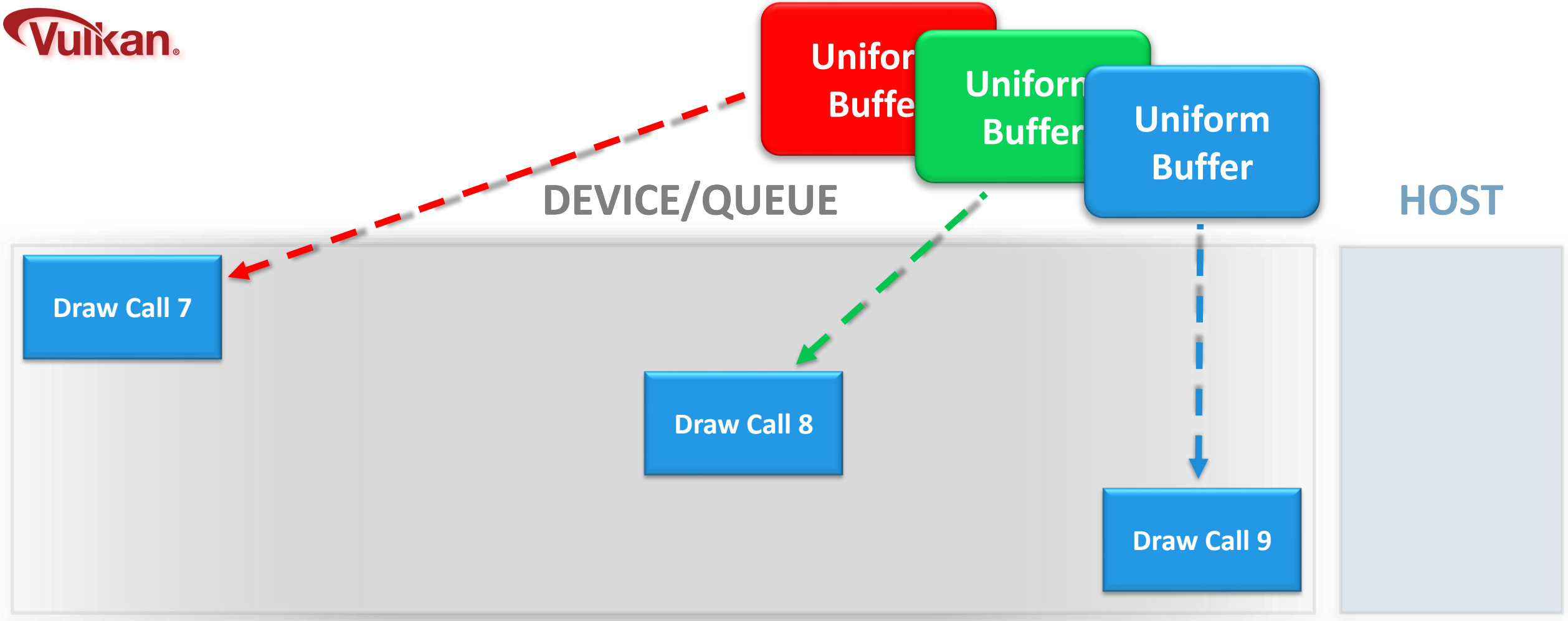
Vulkan Application Configuration



DEVICE/QUEUE

HOST

Vulkan Application Configuration



OpenGL Application Configuration

`glGenBuffers(...)`
bind a named buffer object

`glBindBuffer(GL_UNIFORM_BUFFER, ...)`
generate buffer object names

`glBufferSubData(GL_UNIFORM_BUFFER, ...)`
updates a subset of a buffer object's data store

When replacing the entire data store, consider using **glBufferSubData** rather than completely recreating the data store with **glBufferData**. This avoids the cost of reallocating the data store.

OpenGL Application Configuration

`glGenBuffers(...)`
bind a named buffer object

`glBindBuffer(GL_UNIFORM_BUFFER, ...)`
generate buffer object names

`glBufferSubData(GL_UNIFORM_BUFFER, ...)`
updates a subset of a buffer object's data store

When replacing the entire data store, consider using **glBufferSubData** rather than completely recreating the data store with **glBufferData**. **This avoids the cost of reallocating the data store.**

OpenGL Application Configuration

`glBufferStorage(GL_UNIFORM_BUFFER, ...)`
creates and initializes a buffer object's immutable data store

- `GL_DYNAMIC_STORAGE_BIT`
- `GL_MAP_READ_BIT`
- `GL_MAP_WRITE_BIT`
- `GL_MAP_PERSISTENT_BIT`
- `GL_MAP_COHERENT_BIT`

The client's pointer to the data store remains valid so long as the data store is mapped, even during execution of drawing or dispatch commands.

Copyright © 2014 Khronos Group. OpenGL 4.5 Reference Pages

`glMapBufferRange(GL_UNIFORM_BUFFER, ...)`
map all or part of a buffer object's data store into the client's address space

OpenGL Application Configuration

`glBufferStorage(GL_UNIFORM_BUFFER, ...)`
creates and initializes a buffer object's immutable data store

- `GL_DYNAMIC_STORAGE_BIT`
- `GL_MAP_READ_BIT`
- `GL_MAP_WRITE_BIT`
- `GL_MAP_PERSISTENT_BIT`
- `GL_MAP_COHERENT_BIT`

The client's pointer to the data store remains **valid** so long as the data store is mapped, even **during execution of drawing** or dispatch commands.

Copyright © 2014 Khronos Group. OpenGL 4.5 Reference Pages

`glMapBufferRange(GL_UNIFORM_BUFFER, ...)`
map all or part of a buffer object's data store into the client's address space

OpenGL Application Configuration

Horror Stories

Buffer update

To maintain the illusion, the driver must track resources that are referenced by pending render commands. The driver locks them to prevent modification until those rendering commands have been completed.

If the application attempts to modify a locked resource, then the driver must take some evasive action. Either draining the pipeline until the lock is released, or creating a new ghost copy of the resource to contain the modifications. Both choices incur an overhead that the application can avoid.

Copyright © 2017, 2019–2021 Arm Limited. Arm® Mali™ GPU Best Practices, Version 2.2, Developer Guide

Horror Stories

Buffer update

To maintain the **illusion**, the driver must track resources that are referenced by pending render commands. The driver locks them to prevent modification until those rendering commands have been completed.

If the application attempts to modify a locked resource, then the driver must take some **evasive action**. Either draining the pipeline until the lock is released, or creating a new ghost copy of the resource to contain the modifications. Both choices incur an overhead that the application can avoid.

Copyright © 2017, 2019–2021 Arm Limited. Arm® Mali™ GPU Best Practices, Version 2.2, Developer Guide

101897_0202_01_en

Horror Stories

[…] Either draining the pipeline until the lock is released, […]

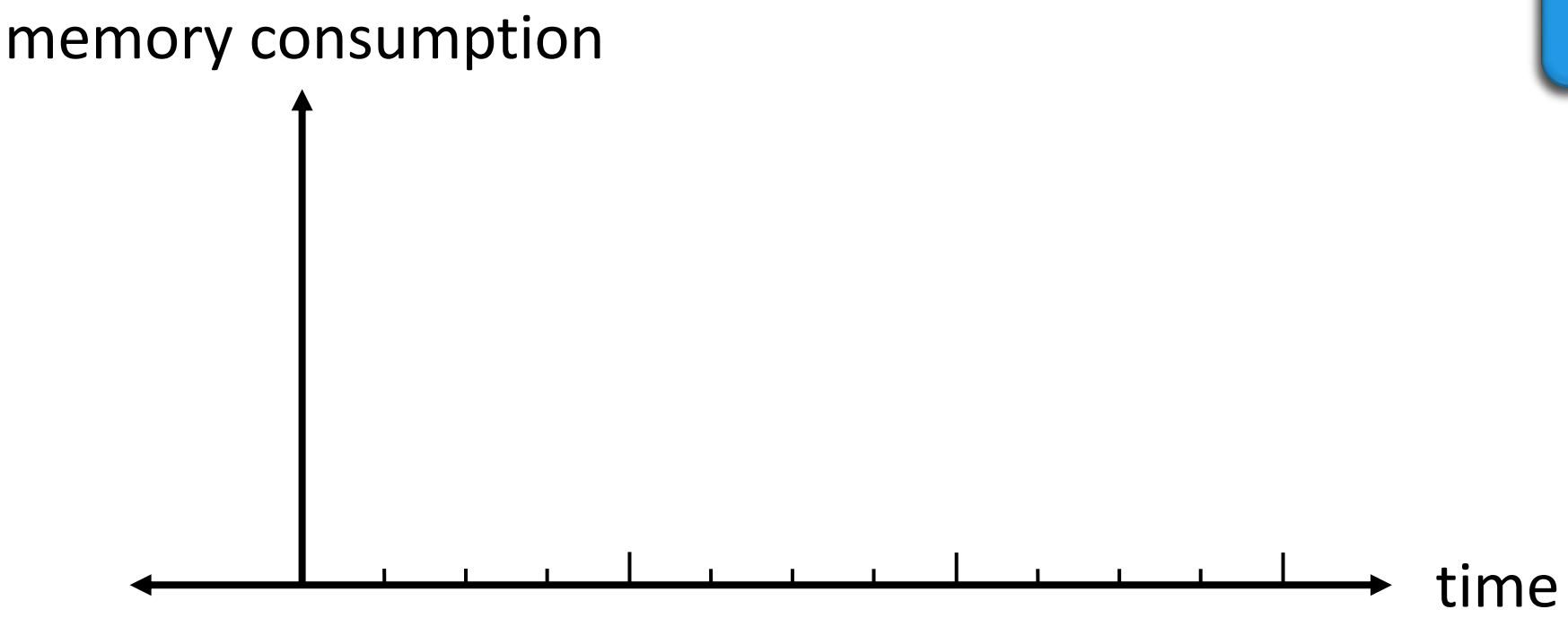
Copyright © 2017, 2019–2021 Arm Limited. Arm® Mali™ GPU Best Practices, Version 2.2, Developer Guide

Horror Stories

[…] or creating a new ghost copy of the resource to contain the modifications. […]

Copyright © 2017, 2019–2021 Arm Limited. Arm® Mali™ GPU Best Practices, Version 2.2, Developer Guide

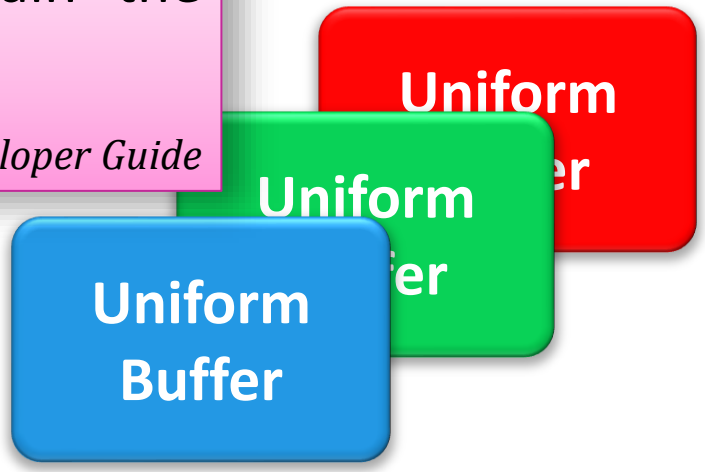
Uniform Buffer

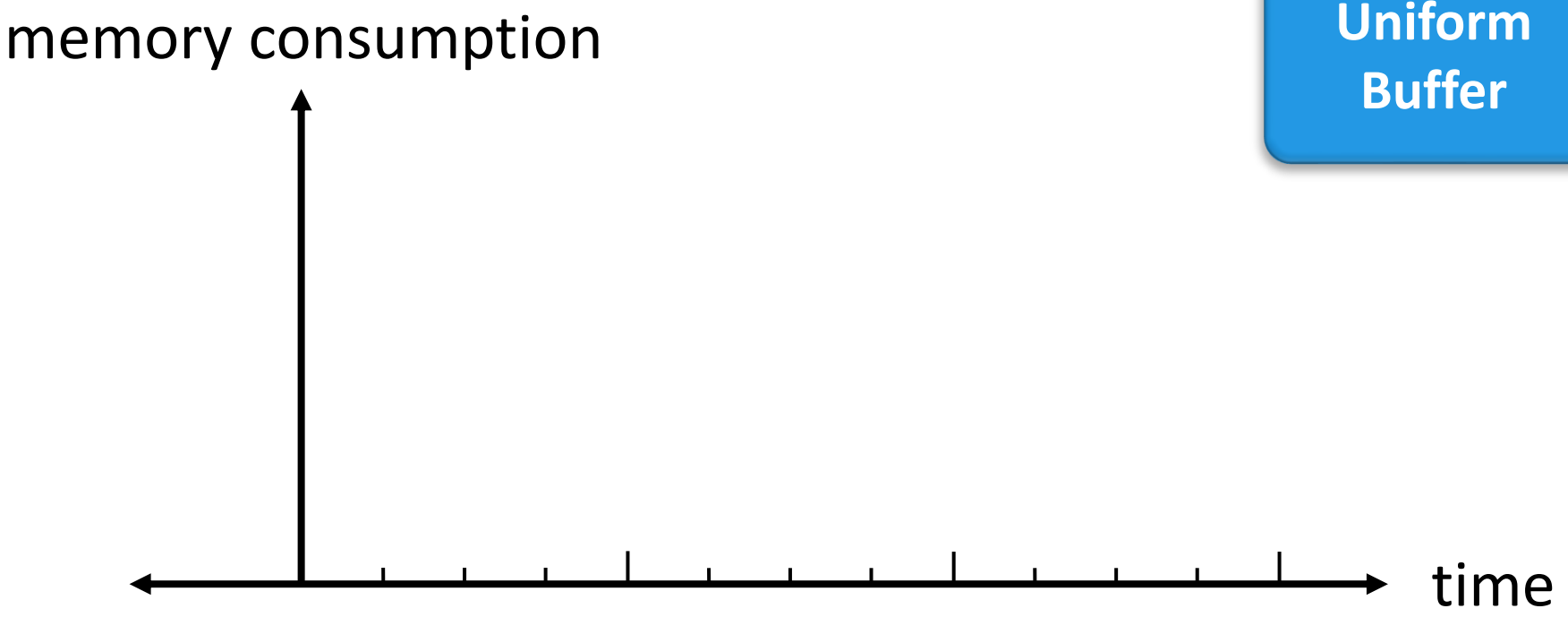


Horror Stories

[…] or creating a new ghost copy of the resource to contain the modifications. […]

Copyright © 2017, 2019–2021 Arm Limited. Arm® Mali™ GPU Best Practices, Version 2.2, Developer Guide

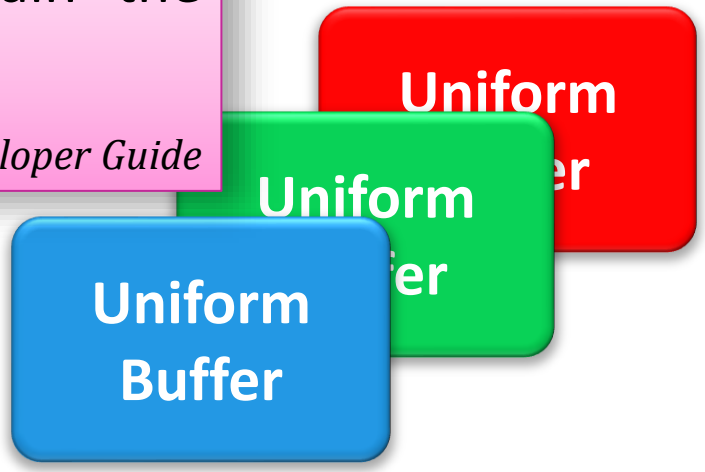


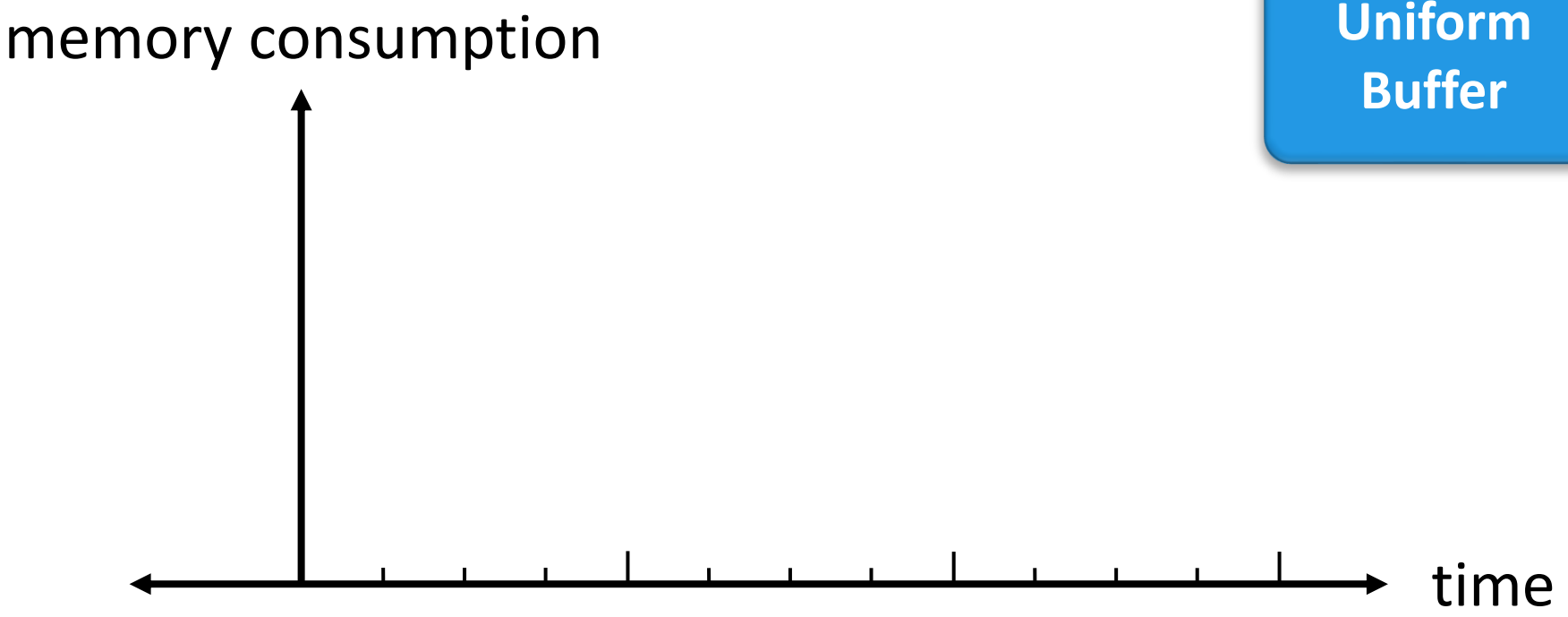


Horror Stories

[…] or creating a new ghost copy of the resource to contain the modifications. […]

Copyright © 2017, 2019–2021 Arm Limited. Arm® Mali™ GPU Best Practices, Version 2.2, Developer Guide





OpenGL Application Configuration

OpenGL Application Configuration

`glBufferStorage(GL_UNIFORM_BUFFER, ...)`
creates and initializes a buffer object's immutable data store

`glMapBufferRange(GL_UNIFORM_BUFFER, ...)`
map all or part of a buffer object's data store into the client's address space

Avoid using `glMapBufferRange()` with either `GL_MAP_INVALIDATE_RANGE`, or `GL_MAP_INVALIDATE_BUFFER`. Both of these flags can trigger the creation of a resource ghost on some Mali driver versions.

Copyright © 2017, 2019–2021 Arm Limited. Arm® Mali™ GPU Best Practices, Version 2.2, Developer Guide

`GL_MAP_PERSISTENT_BIT`

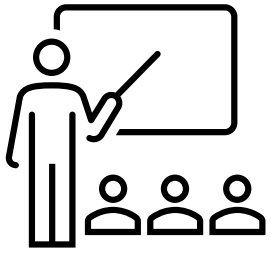
`GL_MAP_COHERENT_BIT`

`GL_MAP_INVALIDATE_RANGE_BIT` ✗

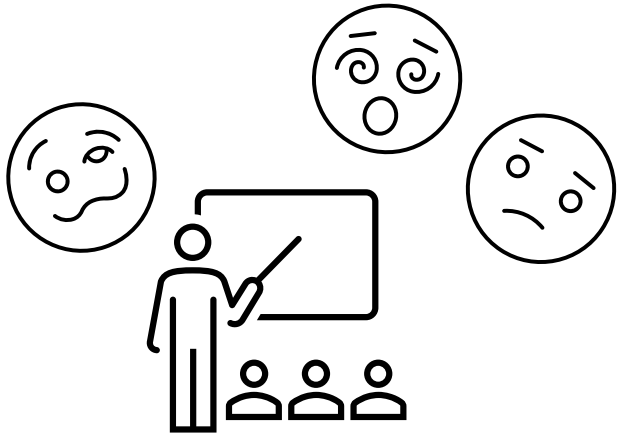
`GL_MAP_UNSYNCHRONIZED_BIT` ✓

Copyright © 2010-2014 Khronos Group. OpenGL 4.5 Reference Pages

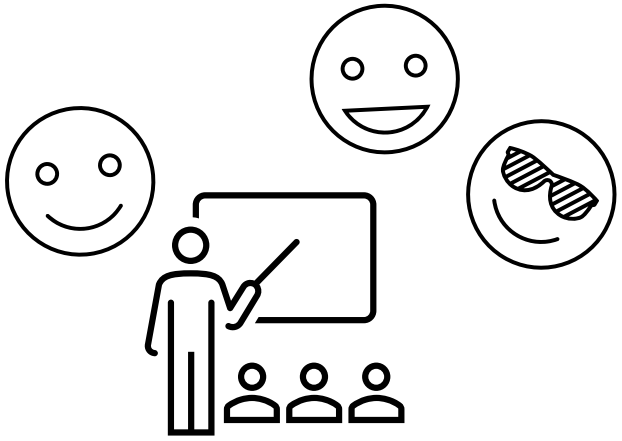
Different Roads To Be Taken



Different Roads To Be Taken



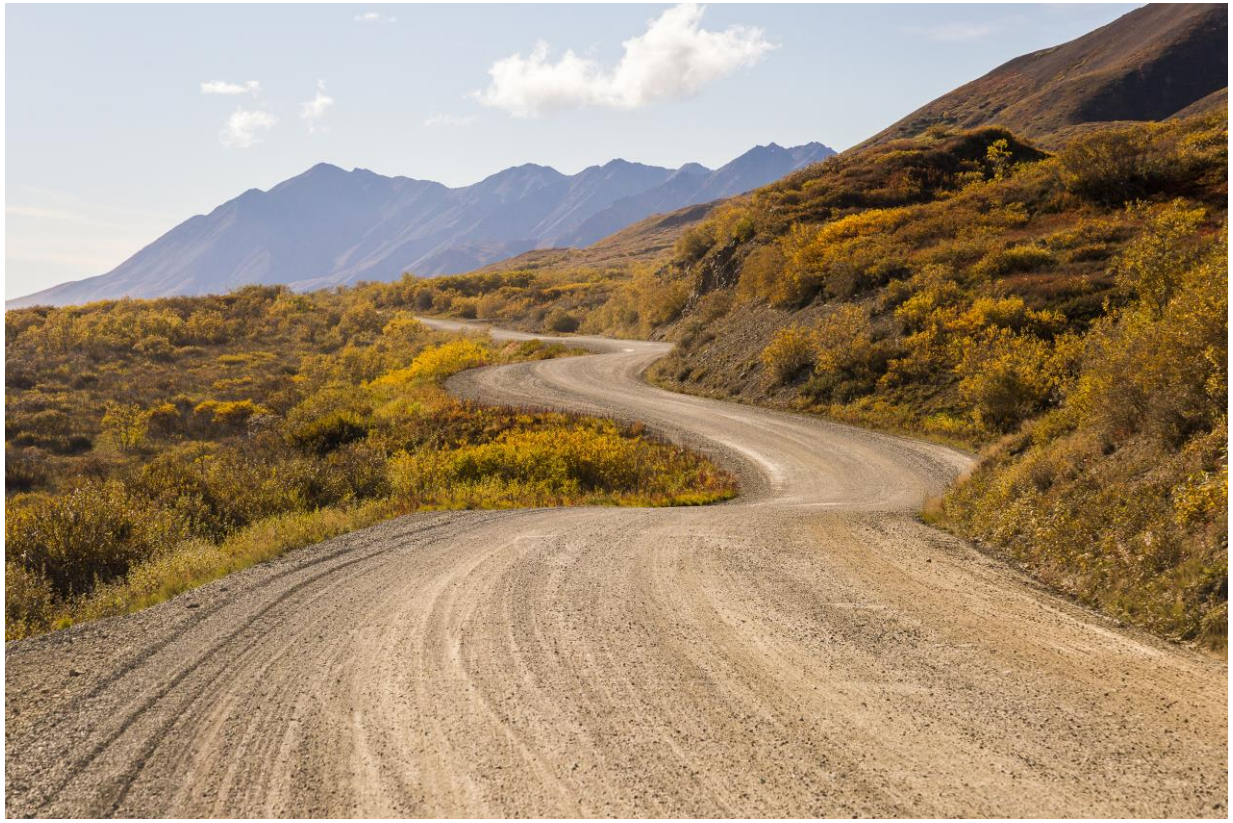
Different Roads To Be Taken



Different Roads To Be Taken

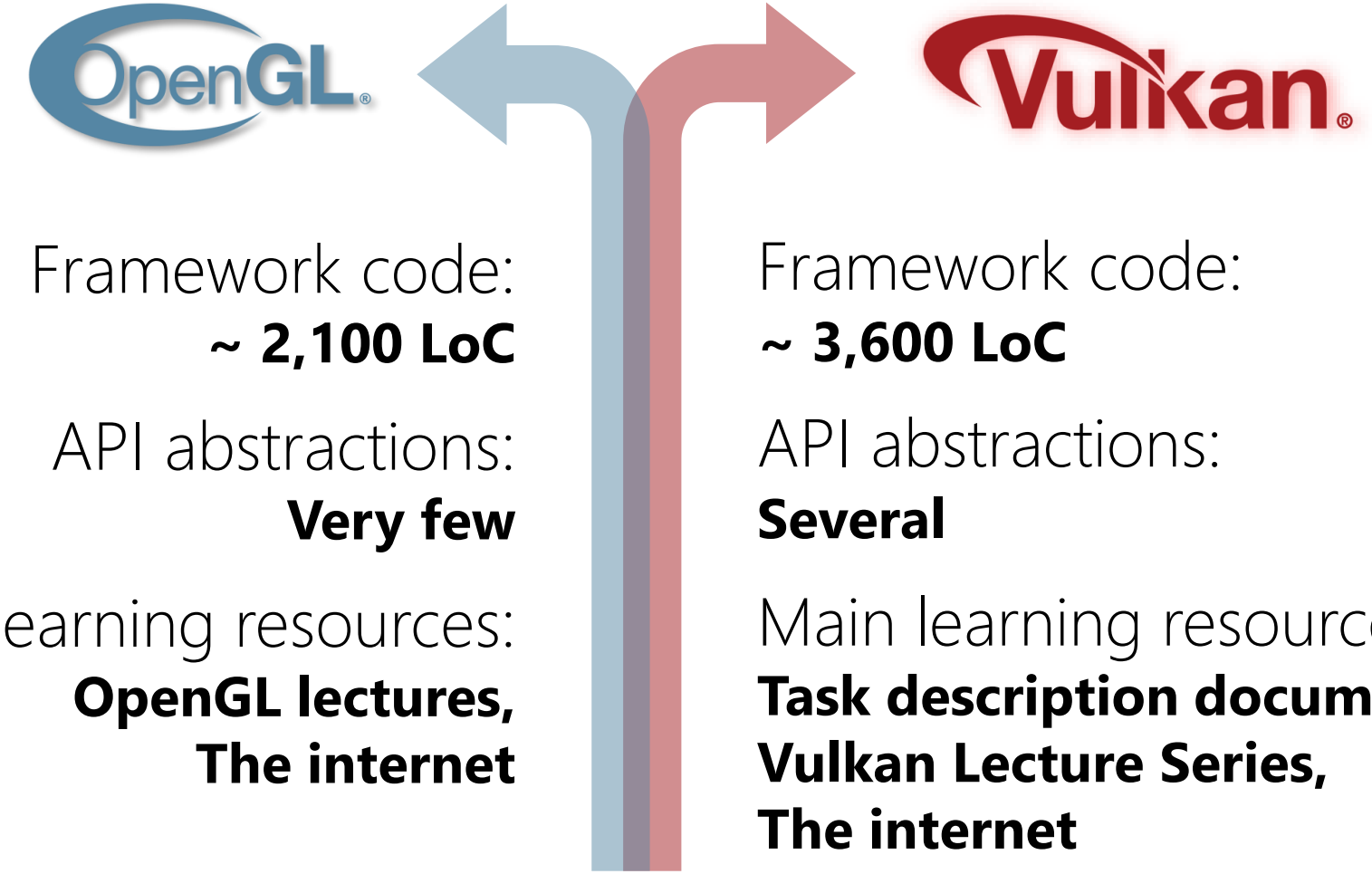
OpenGL®

Vulkan®



Different Approaches in Teaching and Learning

Transition in Introductory Graphics Courses



Framework code:
~ 2,100 LoC

API abstractions:
Very few

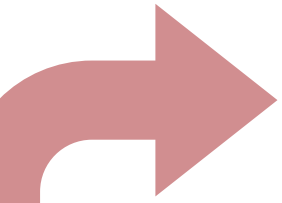
Main learning resources:
OpenGL lectures,
The internet

Framework code:
~ 3,600 LoC

API abstractions:
Several

Main learning resources:
Task description documents,
Vulkan Lecture Series,
The internet

The Road to Vulkan



Framework code:
~ 3,600 LoC

API abstractions:
Several

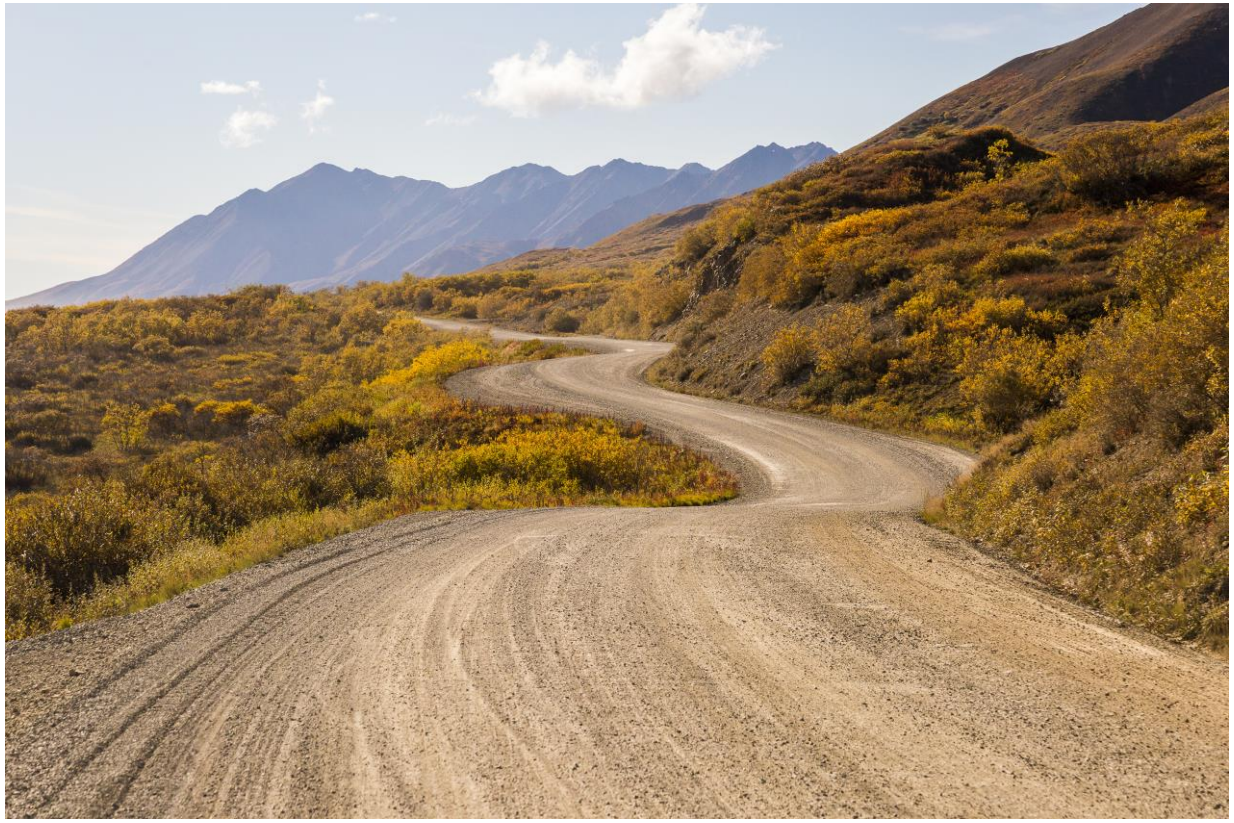
Main learning resources:
Task description documents,
Vulkan Lecture Series,
The internet

The Road to Vulkan

Framework code:
~ 3,600 LoC

API abstractions:
Several

Main learning resources:
Task description documents,
Vulkan Lecture Series,
The internet



The Road to Vulkan

Framework code:
~ 3,600 LoC

API abstractions:
Several

Main learning resources:
Task description documents,
Vulkan Lecture Series,
The internet

Assignment 1: Abstract swap chain handling and its synchronization, abstract render pass creation, abstract framebuffer creation

Assignment 2: Abstract parts of graphics pipeline creation, abstract memory management, abstract command buffers

Assignment 3: Re-introduce command buffer recording

Assignment 5: Introduce synchronization, introduce image layout transitions, introduce device memory (usage)

The Road to Vulkan

Framework code:
~ 3,600 LoC

API abstractions:
Several

Main learning resources:
Task description documents,
Vulkan Lecture Series,
The internet

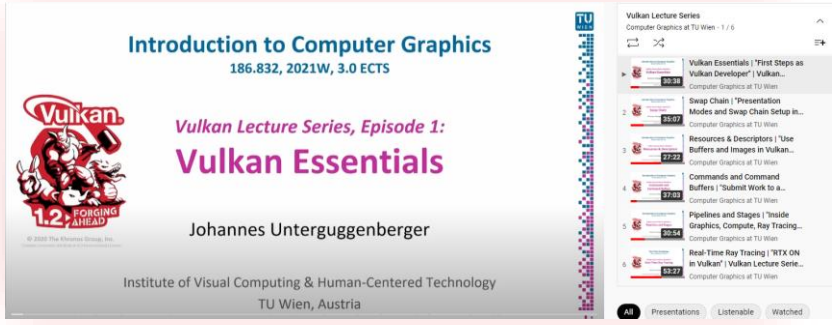
Assignment 1: Abstract swap chain handling and its synchronization, abstract render pass creation, abstract framebuffer creation

Assignment 2: Abstract parts of graphics pipeline creation, abstract memory management, abstract command buffers

Assignment 3: Re-introduce command buffer recording

Assignment 5: Introduce synchronization, introduce image layout transitions, introduce device memory (usage)

Vulkan Lecture Series



The Road to Vulkan

Framework code:
~ 3,600 LoC

API abstractions:
Several

Main learning resources:
Task description documents,
Vulkan Lecture Series,
The internet

Assignment 1: Abstract swap chain handling and its synchronization, abstract render pass creation, abstract framebuffer creation

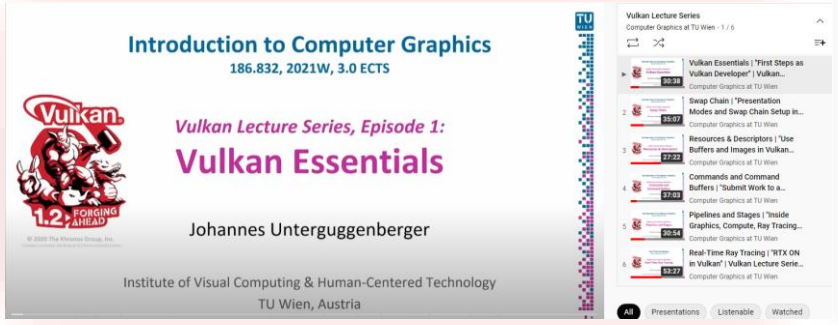
Assignment 2: Abstract parts of graphics pipeline creation, abstract memory management, abstract command buffers

Assignment 3: Re-introduce command buffer recording

Assignment 5: Introduce synchronization, introduce image layout transitions, introduce device memory (usage)

Computer Graphics at TU Wien
1.37K subscribers

Vulkan Lecture Series



Takeaway Messages

The Road to Vulkan®

Different road for teaching/learning
graphics programming

You are in the driver's seat
(not your GPU's driver)

Totally possible in introductory graphics courses.
See *our paper* for more details and student feedback:

The Road to Vulkan:
Teaching Modern Low-Level APIs in Introductory Graphics Courses

Johannes Unterguggenberger
Bernhard Kerbl
Michael Wimmer

Thank You
for Your Attention

Transition in Introductory Graphics Courses