Nathan Gauër - 2018

3D Acceleration on Windows, Part II

VirtIO-GPU, Windows and Angle

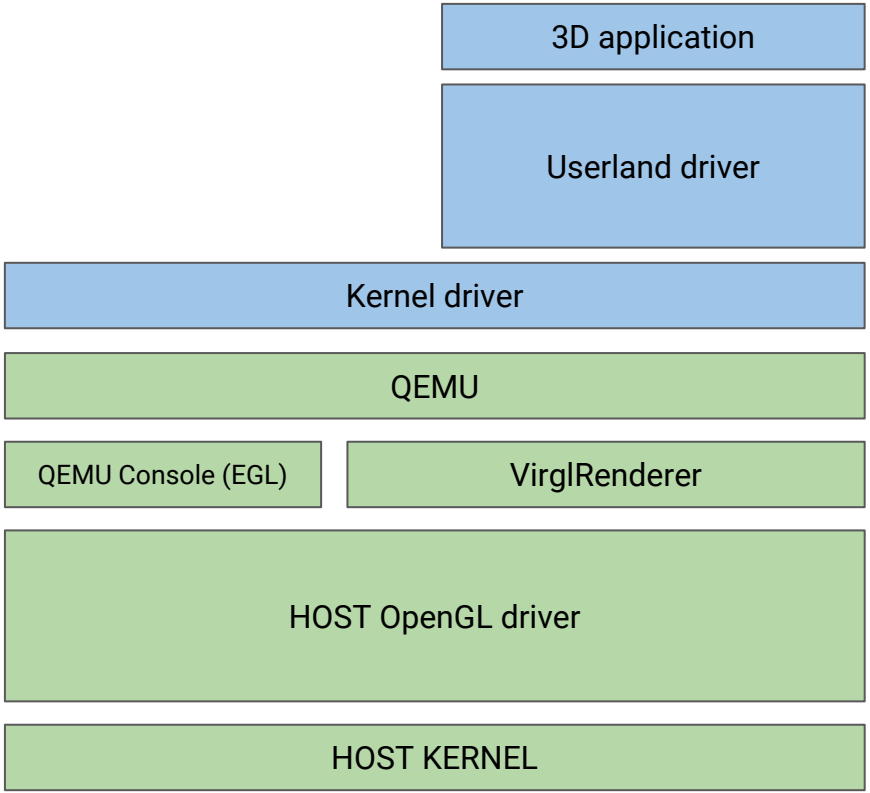
Context

VirtioGPU & Linux

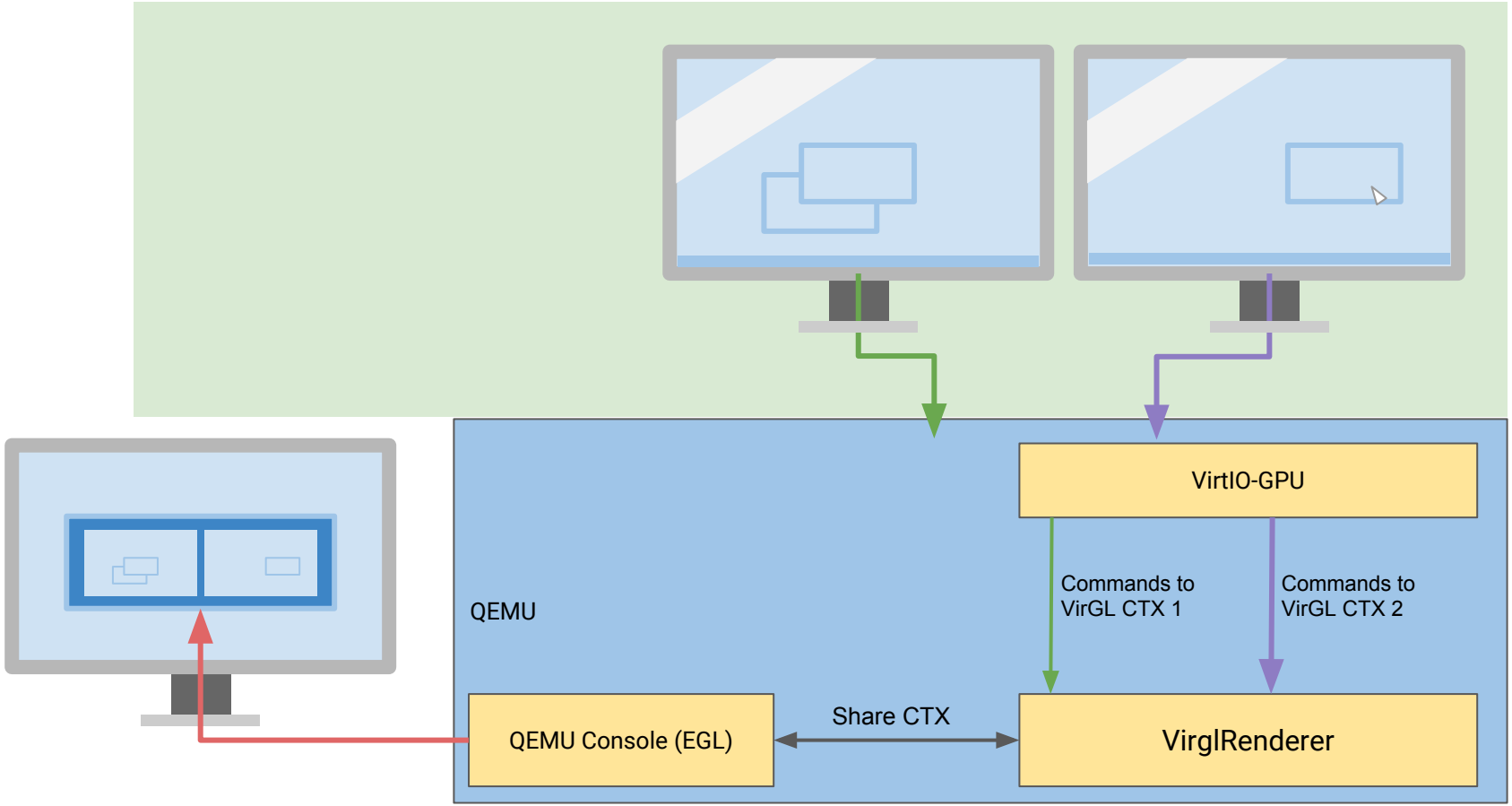
Have HW acceleration on a guest

Working on Linux

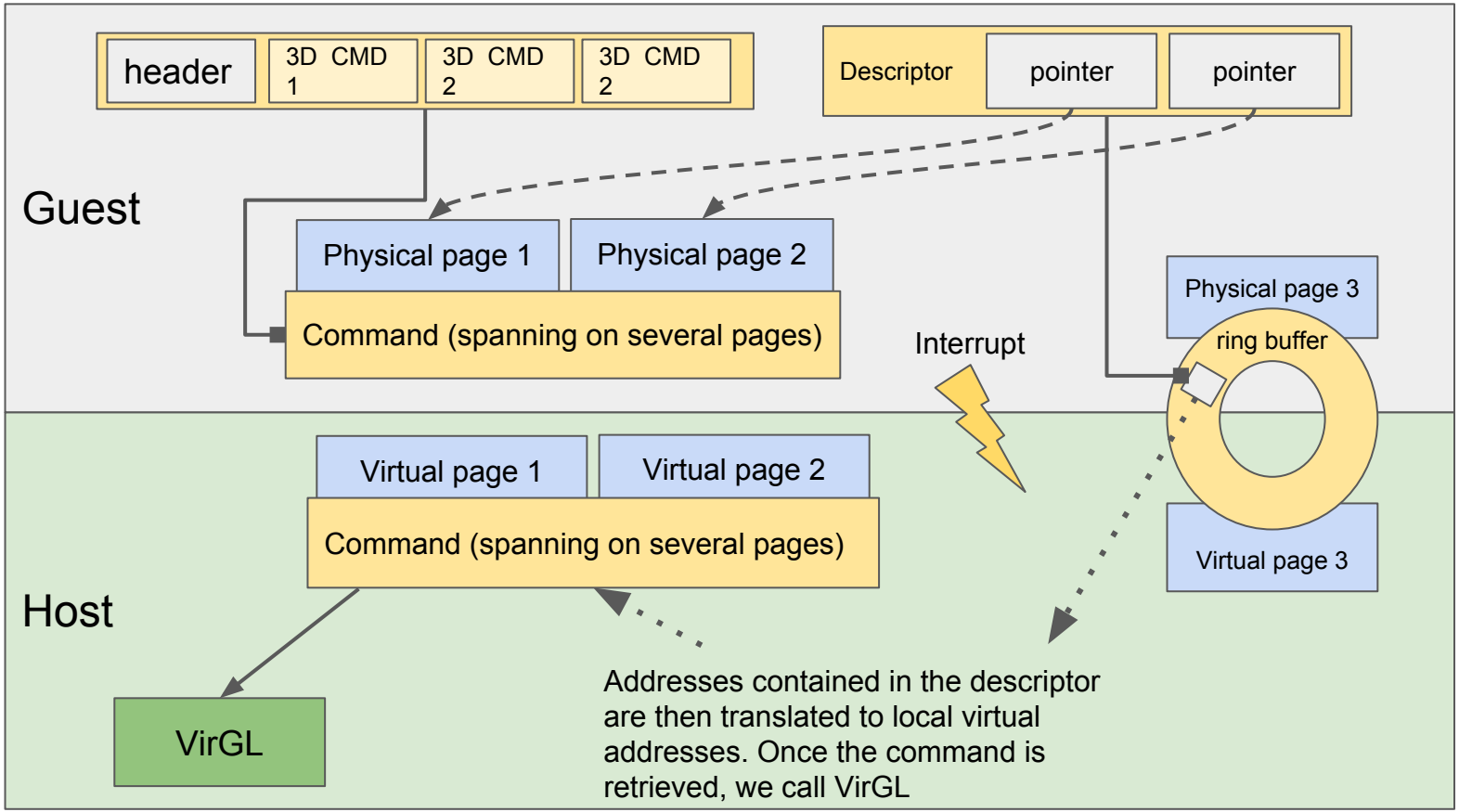
Using MESA



VirtioGPU

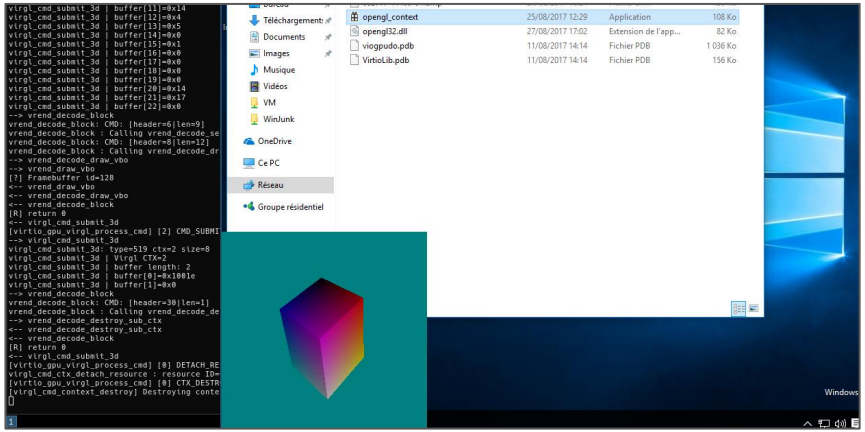

VirtIO-GPU
Commands to
VirGL CTX 1
Commands to
VirGL CTX 2
QEMU
QEMU Console (EGL)
Share CTX
VirglRenderer

VirtioGPU


header
3D CMD 1
3D CMD 2
3D CMD 2
Descriptor
pointer
pointer
Guest
Physical page 1
Physical page 2
Command (spanning on several pages)
Physical page 3
ring buffer
Interrupt
Virtual page 1
Virtual page 2
Command (spanning on several pages)
Virtual page 3
Host
VirGL
Addresses contained in the descriptor are then translated to local virtual addresses. Once the command is retrieved, we call VirGL

VirtioGPU & Windows

- Basic PoC
- Home-made ‘state-tracker’
- No Windows compositor integration
- Userland is trusted
- Userland dictates kernel allocations



Output using VirtioGPU PoC on QEMU

What

Goals

Have a valid usermod + kernel driver

Support OpenGLES 2.0

Support Vulkan

On Windows

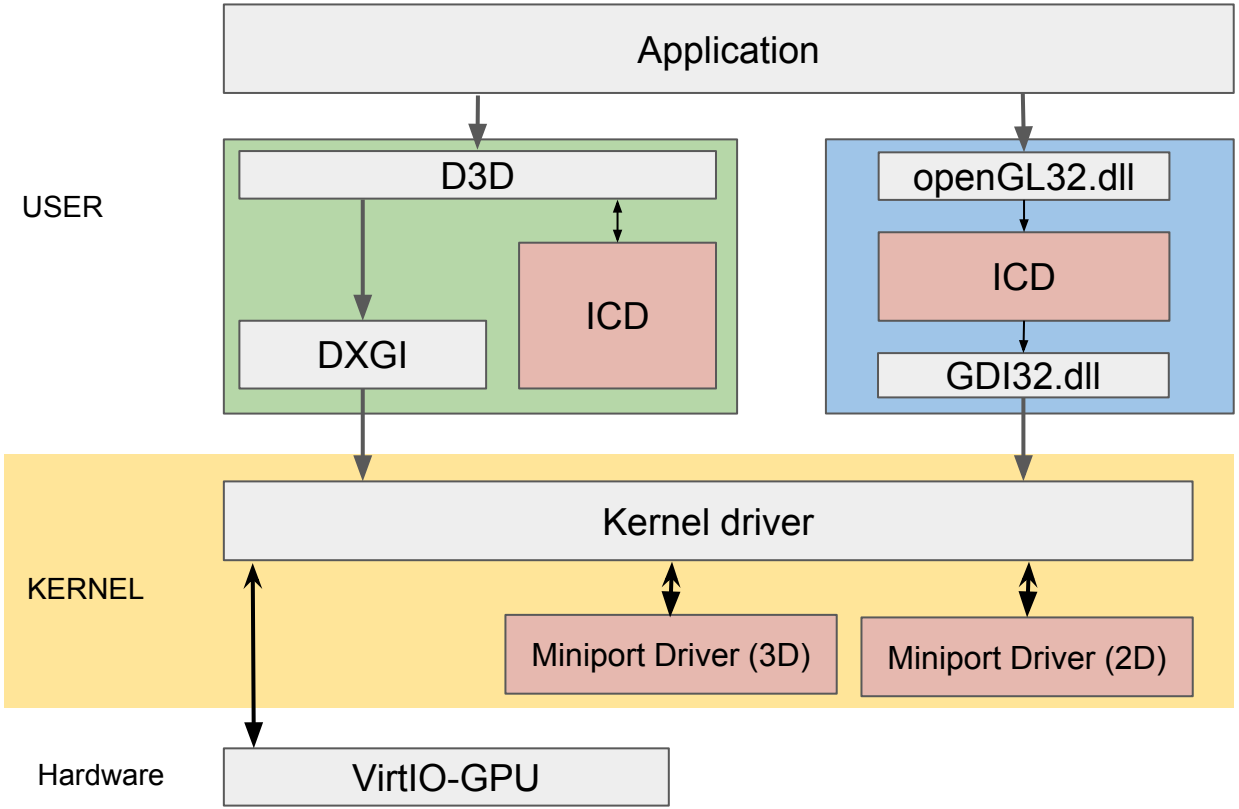


figure: windows graphic stack

On Windows

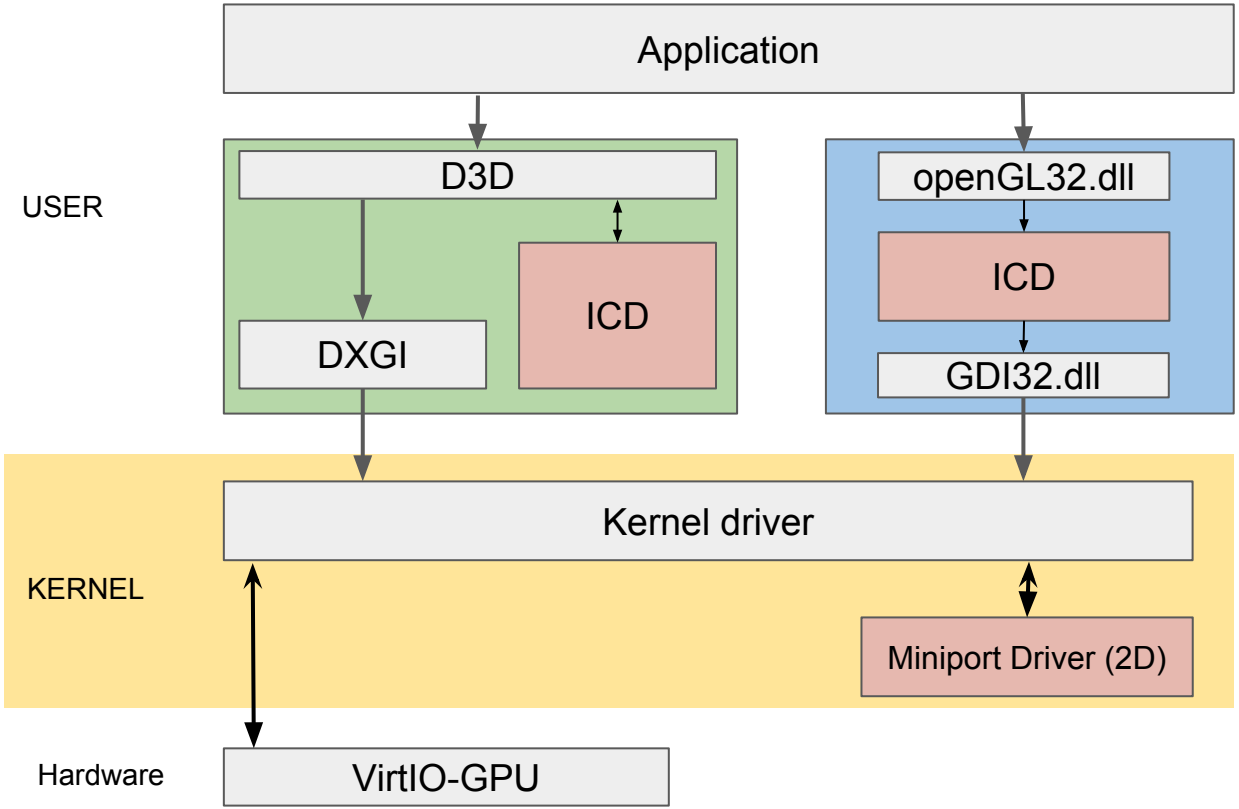


figure: windows graphic stack

How

MESA

Use existing GL state-tracker

Create Winsys for Virtio GPU

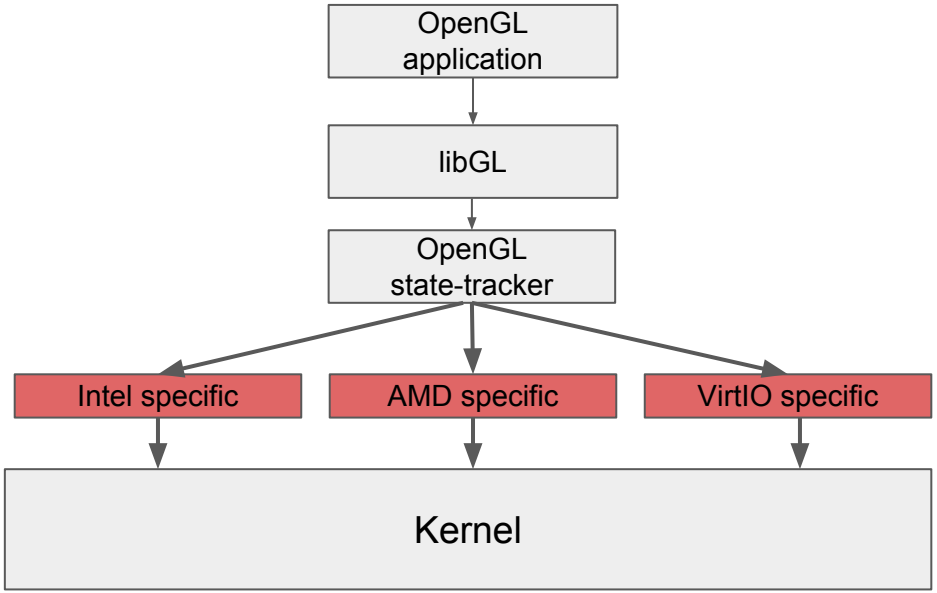


figure: MESA winsys concept

MESA

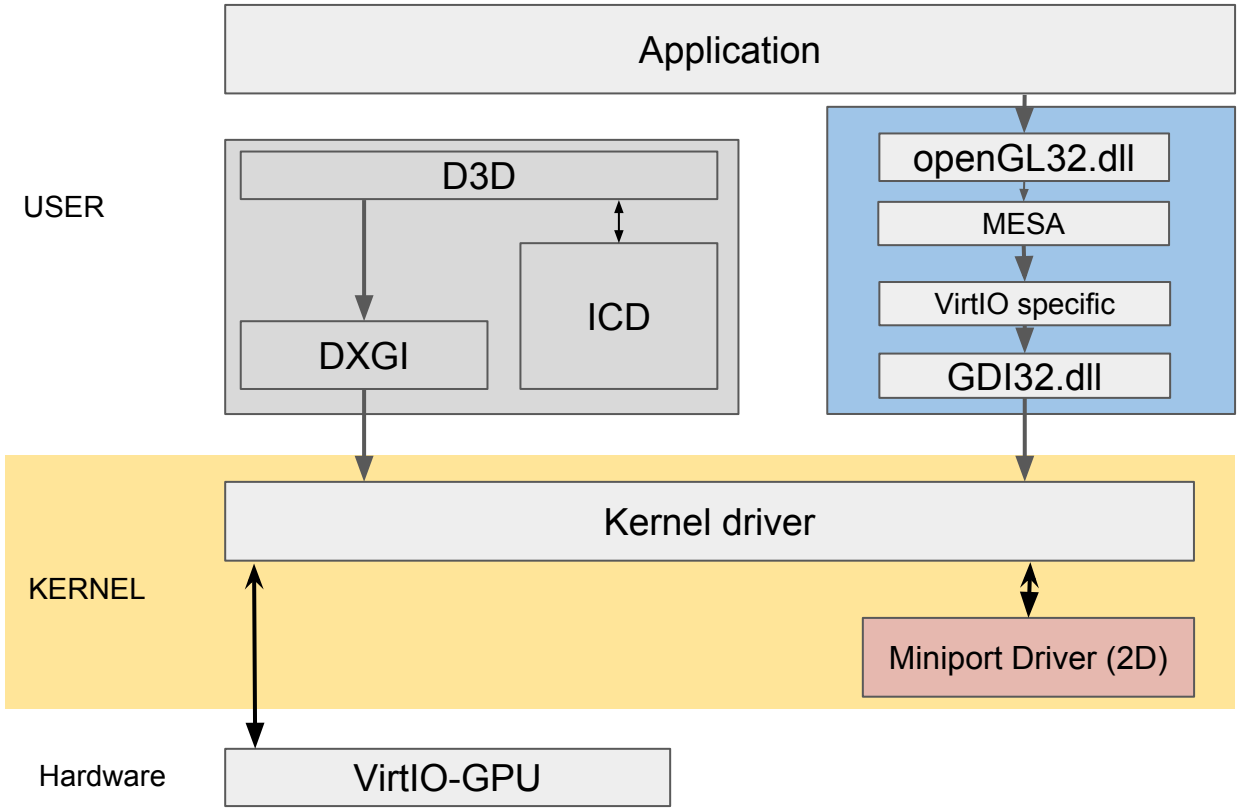


figure: windows graphic stack using MESA

Google's *"Google traduction"* for graphic APIs

IN \ OUT	DirectX 9	DirectX 11	OpenGL	GL ES	Vulkan
OpenGL ES 2	yes	yes	yes	yes	in progress
OpenGL ES 3		yes	yes	in progress	no

Supported backends by frontend (on windows)

ANGLE

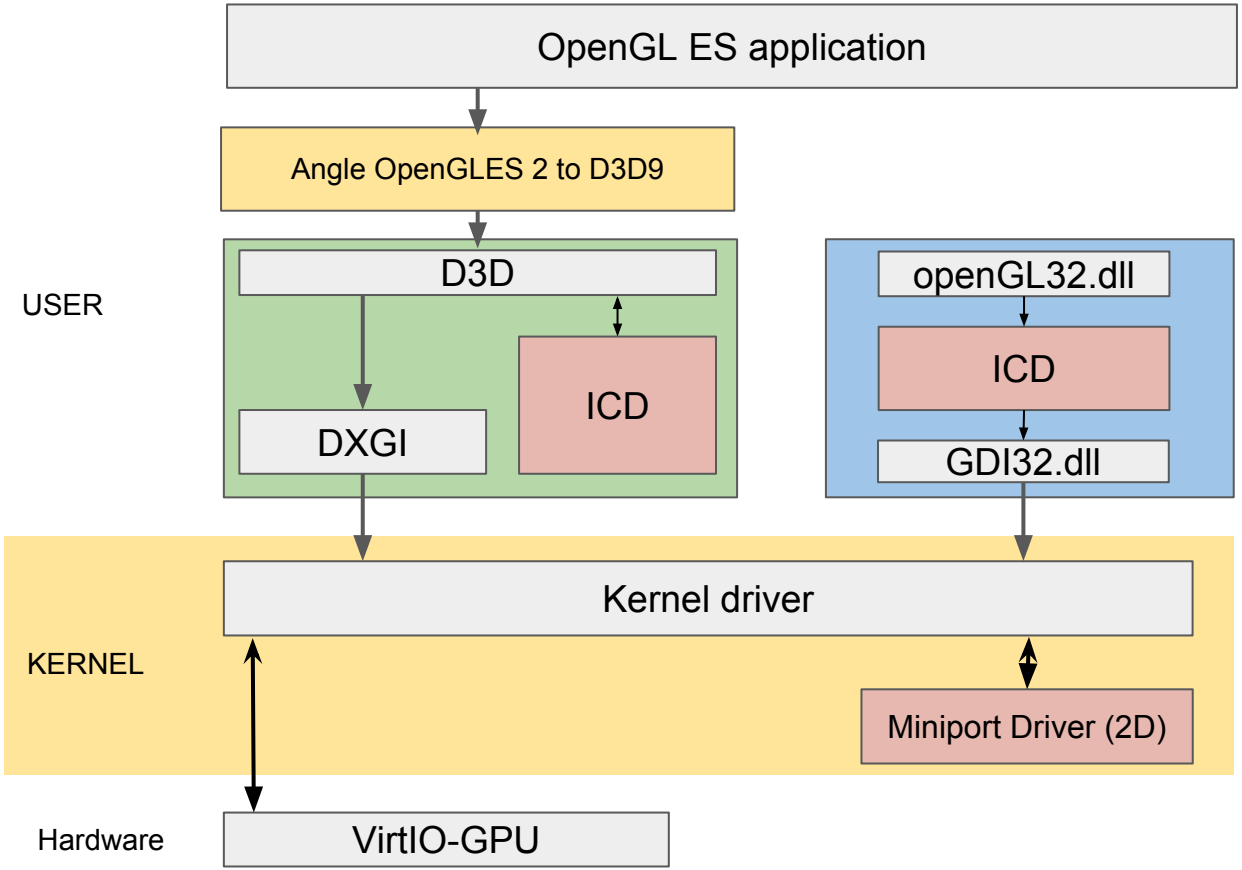


figure: windows graphic stack using Angle

ANGLE

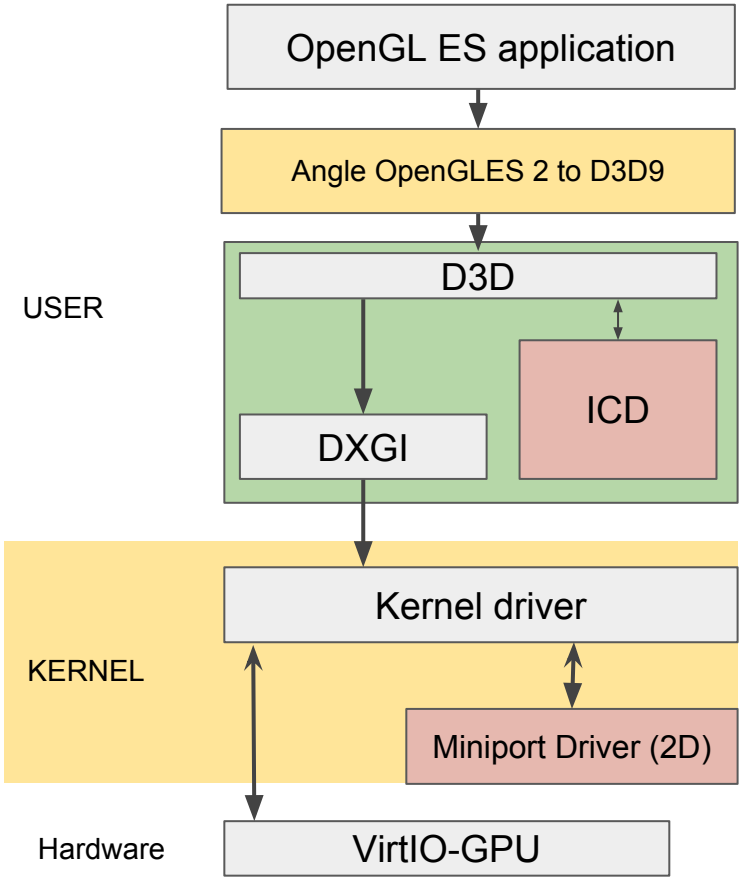


figure: windows graphic stack using Angle

Questions ?

https://github.com/google/angle

https://github.com/vrozenfe/virtio-gpu-win

https://docs.microsoft.com/en-us/windows-hardware/drivers/display/windows-vista-display-driver-model-design-guide