OPENGL SUPPORT IN EASYBUILD

libglvnd as a vendor-neutral dispatch layer for arbitrating OpenGL API

2025-03-26 I JENS H. GÖBBERT (J.GOEBBERT@FZ-JUELICH.DE)

WHO I AM

- **Jülich Supercomputing Centre**
  - Division “Application Support”
    - Algorithms, Tools and Methods Lab (ATML) "Visualization & Interactive HPC“

- **Expertise in**
  - Interactive high-performance computing
  - Visualization of large scientific data sets
  - In-situ visualization & coupling

- **Passion for**
  - Cloud-HPC coupling
  - JupyterLab
  - ParaView / Catalyst

- **EasyBuild**
  - Part of the software team @ JSC
  - Build EasyConfigs for JSC (since 2020)
  - Responsible for >100 modules @ JSC

Background

- Started with an Amiga 500 (1990)
- Studied mechanical engineering (until 2006)
- Got into High Performance Computing (2008)
- Started with In-Situ Visualization (2012)
- Focused on Scientific Visualization (2015)
- Permanent staff at the JSC (since 2016)

JÜLICH Forschungszentrum

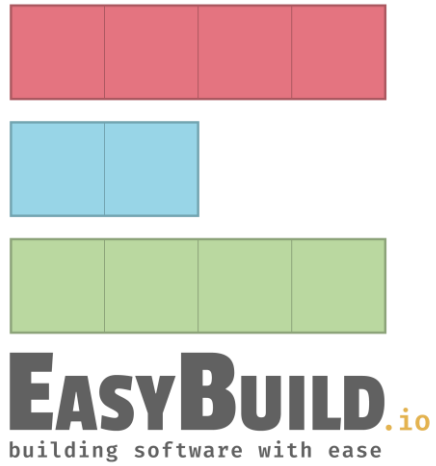
MOTIVATION

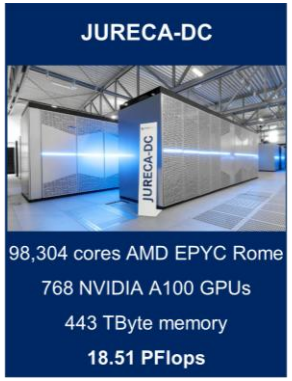
Get most out of the available hardware

Responsible primarily for ebconfigs of

- Visualization software - e.g. ParaView, VisIt, Ascent, VMD
- JupyterLab ecosystem and other WebUIs

and its dependencies.





Obviously (In-Situ)-Visualization tasks can benefit from **hardware-accelerated OpenGL** when rendering

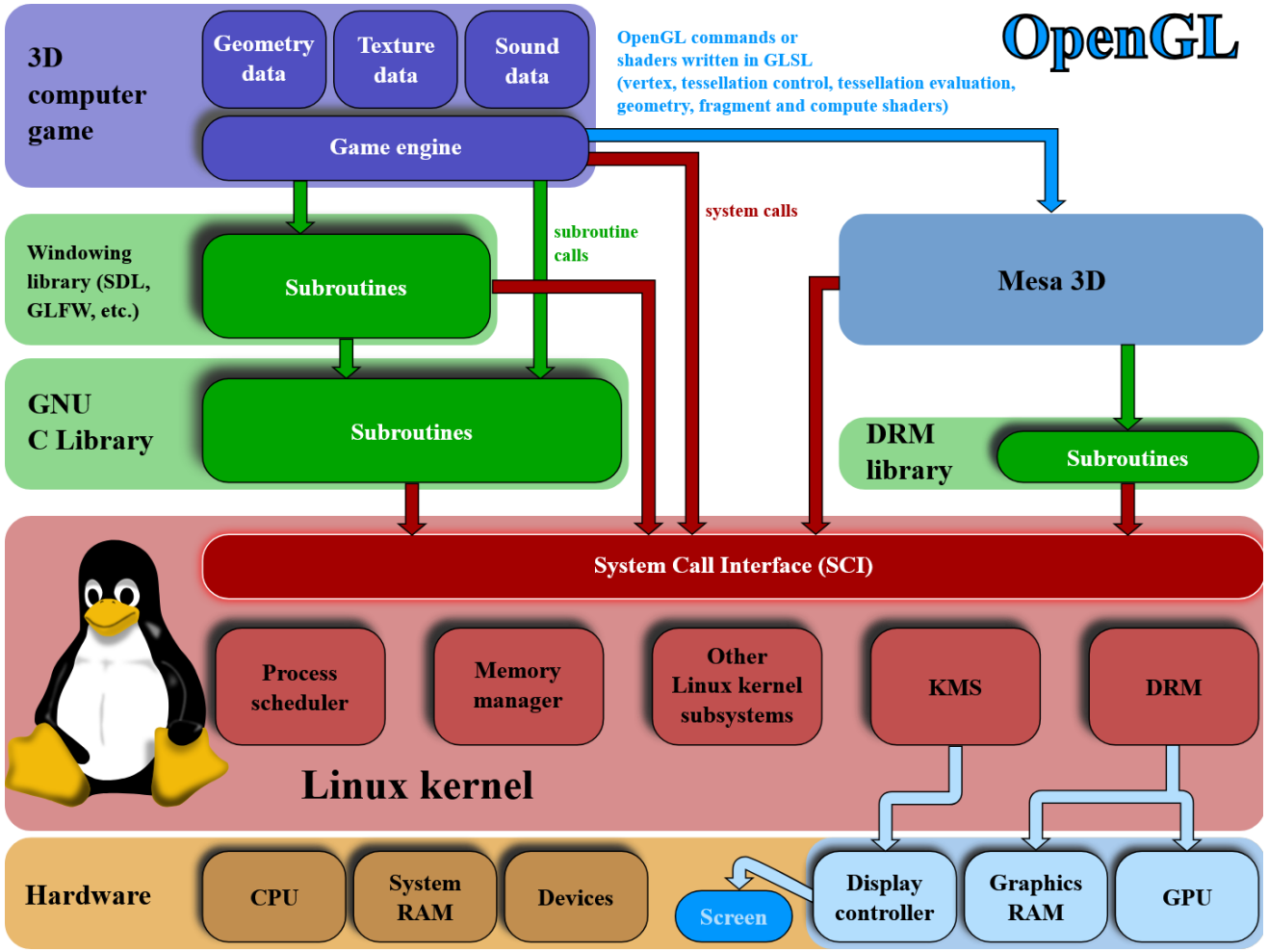
- JUWELS Cluster: 224 NVIDIA V100 GPUs
- JUWELS Booster: 3,744 NVIDIA A100 GPUs
- JURECA-DC: 768 NVIDIA A100 GPUs, 24 NVIDIA Quadro RTX8000
- JUSUF: 45 NVIDIA V100 GPUs

What about JEDI/JUPITER? NVIDIA H200 does not support hardware-accelerated OpenGL ☹

JÜLICH Forschungszentrum

OUTLINE

- Who am I / Motivation
- OpenGL / MESA
- Vendor-neutral dispatch library
- Easybuild solution at JSC
- Summary

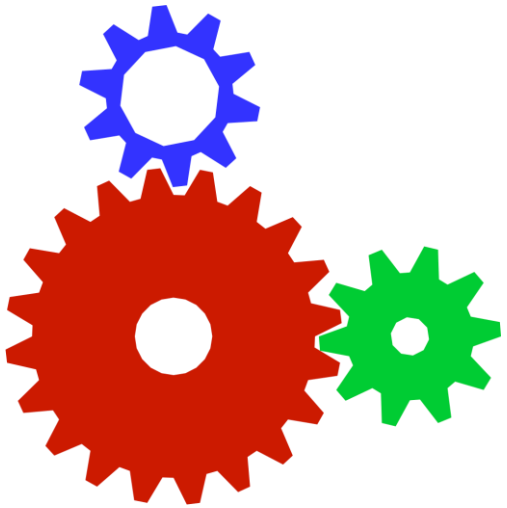


https://en.wikipedia.org/wiki/Mesa_(computer_graphics)

JÜLICH Forschungszentrum

MESA

Software Structure



- **Mesa**, also called **Mesa3D** (initiated in 1993 and hosted by freedesktop.org) is an open source implementation of graphics APIs.

- The primary API is **OpenGL**
  but there's also support for OpenGL ES, Vulkan, **EGL**, OpenCL, and acceleration APIs.

- Mesa translates these specifications to **vendor-specific graphics hardware drivers** [1].
  - Intel, AMD, **NVIDIA (Nouveau),** and others

[1] https://docs.mesa3d.org/systems.html

JÜLICH Forschungszentrum

OPENGL

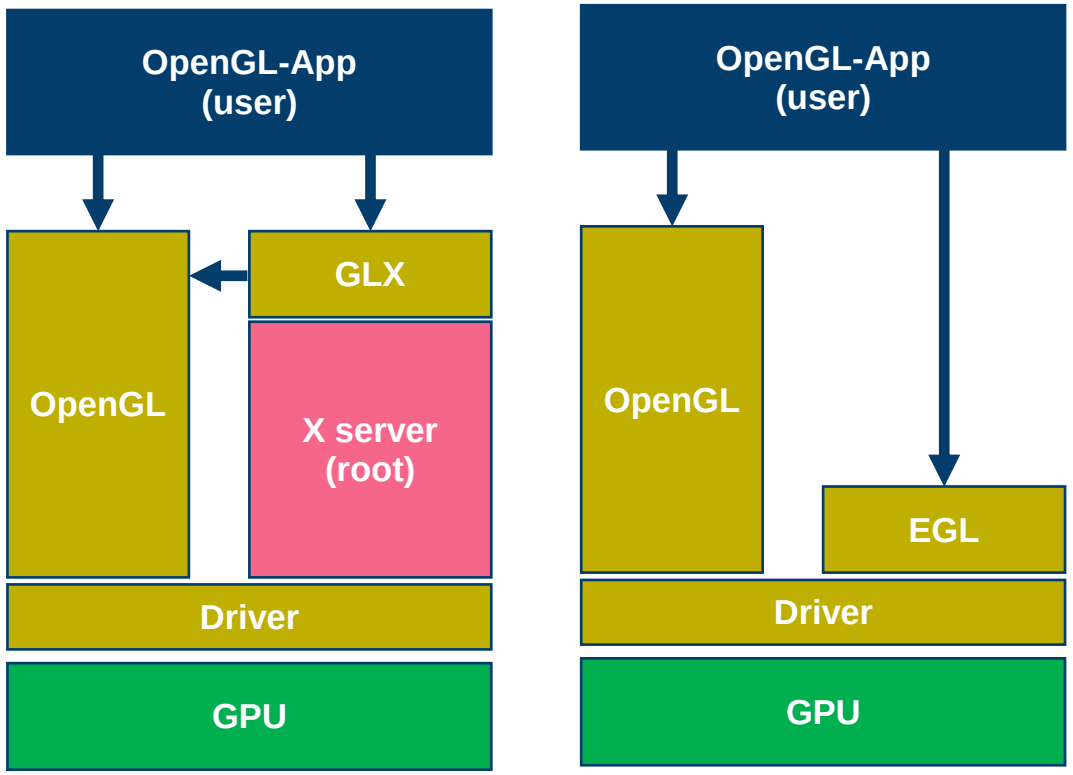
GLX vs. EGL

The GLX extension was created to make OpenGL usable for X applications in a network-transparent manner.

Prior to EGL, an application obtained an OpenGL context by calling the X server via the GLX library.

But X has to run as a privileged process to get full access to the GPU.

EGL simplifies this architecture, by allowing applications to obtain an OpenGL context without an X server.



Comparison of the software stack without EGL (left) and with EGL (right).

JÜLICH
Forschungszentrum

MESA‘S NOUVEAU NOT AN OPTION

NVIDIA’s proprietary graphics drivers do not like MESA

- NVIDIA‘s GPU drivers are a **hard requirement**
  for expensive HPC systems powered by NVIDIA GPUs.

  But,

  the open-source MESA driver Nouveau **cannot co-exist**
  side-by-side with NVIDIA‘s proprietary graphics drivers.

⇒ Hence,

Nouveau is generally **not loaded** by the kernel on an HPC system with NVIDIA GPUs

Proprietary graphics drivers (e.g., NVIDIA driver and AMD Catalyst)
replace all of Mesa, providing their own implementation of a graphics API.

JÜLICH
Forschungszentrum

MESA – SOFTWARE OPENGL

Fall-back solution: Mesa's software-based OpenGL implementations

Mesa also contains **software-based OpenGL implementations**:

Classic:

- swrast (dropped) – first to allow shaders to run on the CPU as a fallback
  - Now `gallium-drivers=swrast`
    is a (deprecated) alias for `gallium-drivers=softpipe,llvmpipe`

Gallium software rasterizer:

- ***llvmpipe*** – generates CPU code at runtime using LLVM
- *softpipe* - on an obscure platform lacking proper LLVM support
- OpenSWR (dropped) - Intel Rasterizer with focus on Advanced Vector Extensions

1) MESA Nouveau is NOT an option and 2) software-based OpenGL is SLOW.
Is there a solution?

LIBGLVND (1)

Vendor-Neutral GL Dispatch Library

https://gitlab.freedesktop.org/glvnd/libglvnd
https://www.youtube.com/watch?v=CxiGE7wcZdE

September 2012:

- Andy Ritger of NVIDIA proposed a revised Linux OpenGL ABI.
- Both **GLX and EGL** are supported, in any combination with OpenGL and OpenGL ES.

Advantages

- Allows multiple vendor implementations to co-exist on the file system.
- And even, allows multiple vendor implementations to co-exist within the same process.

Releases

- libglvnd 0.1.1 - August 2016
- libglvnd 1.0.0 - November 2017
- libglvnd 1.1.0 - August 2018
- libglvnd 1.1.1 - March 2019
- libglvnd 1.2.0 - September 2019
- libglvnd 1.3.0 - Dezember 2019
- libglvnd 1.4.0 - Dezember 2021
- libglvnd 1.5.0 - August 2022
- libglvnd 1.6.0 - November 2022
- **libglvnd 1.7.0 - September 2023**

JÜLICH Forschungszentrum

LIBGLVND (2)

Vendor-Neutral GL Dispatch Library

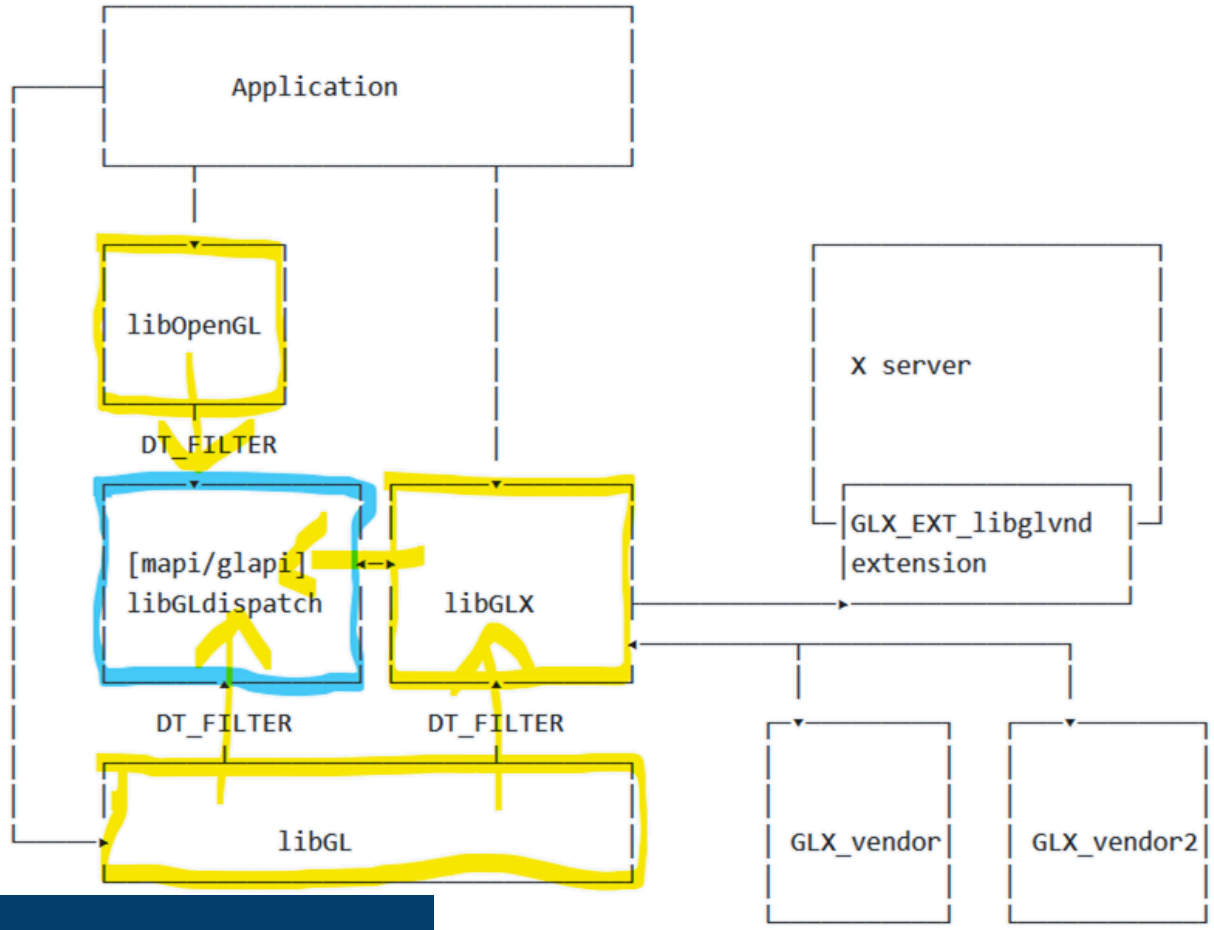
https://gitlab.freedesktop.org/glvnd/libglvnd

Wrapper libraries to libGLdispatch

- libOpenGL
  exposes OpenGL 4.5 core
  and compatibility entry points.
- libGLESv{1,2}
  expose OpenGL ES entrypoints.

Backwards-compatibility

- libGL provided for backwards-compatibility with applications which link against the old ABI



The same OpenGL function in libOpenGL, libGLES, libGL are all interchangeable.

DT_FILTER :
symbols exported by A
are resolved to entrypoints in B

LIBGLVND (3)

Vendor-Neutral GL Dispatch Library

https://gitlab.freedesktop.org/glvnd/libglvnd

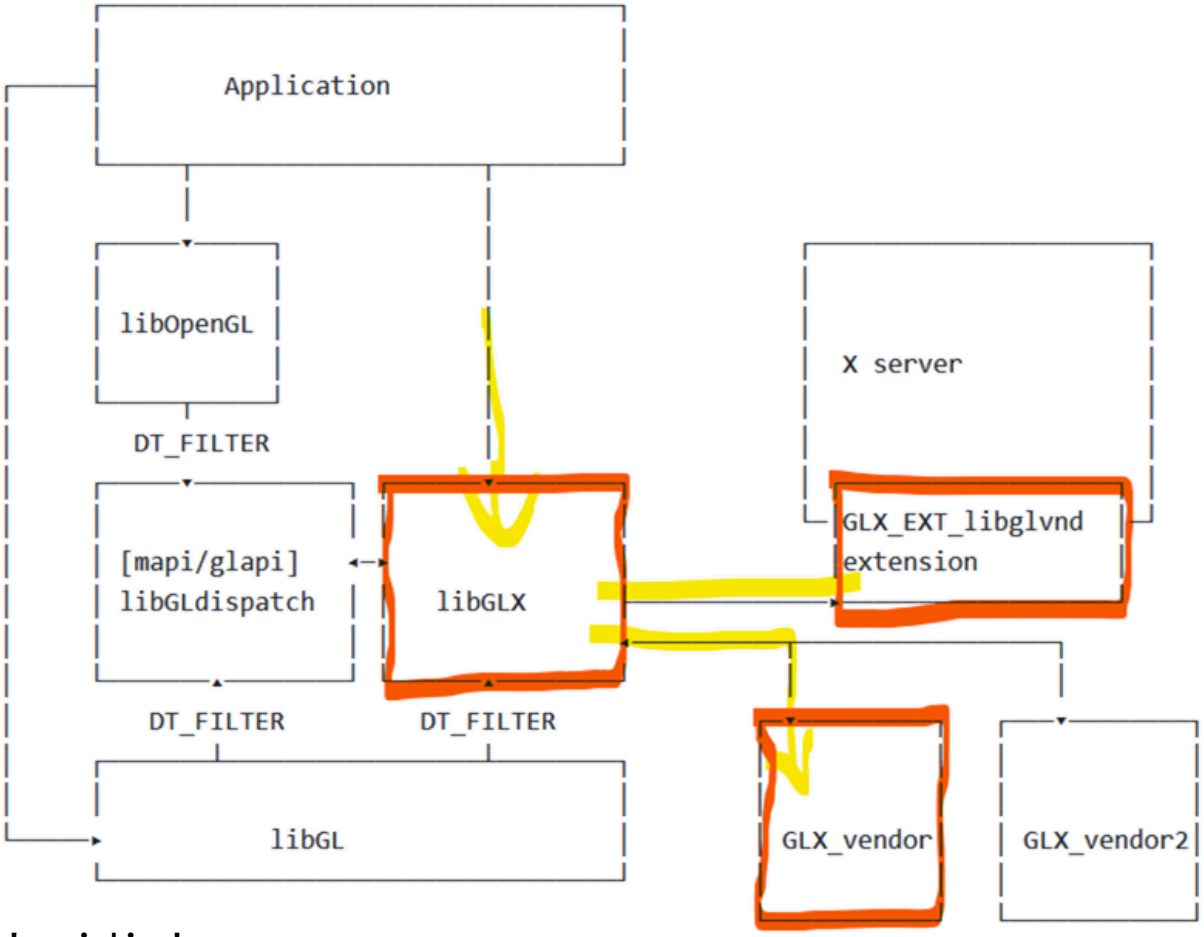
GLX dispatching

- Difficult as many GLX functions are context-independent

GLX dispatching is based on the X screen.
GLX_EXT_libglvnd is supporting here.

This can be overwritten by
__GLX_VENDOR_LIBRARY_NAME

```
export __GLX_VENDOR_LIBRARY_NAME=mesa  # 'mesa', 'nvidia'
```



DT_FILTER :
symbols exported by A
are resolved to entrypoints in B

LIBGLVND (3)

Vendor-Neutral GL Dispatch Library

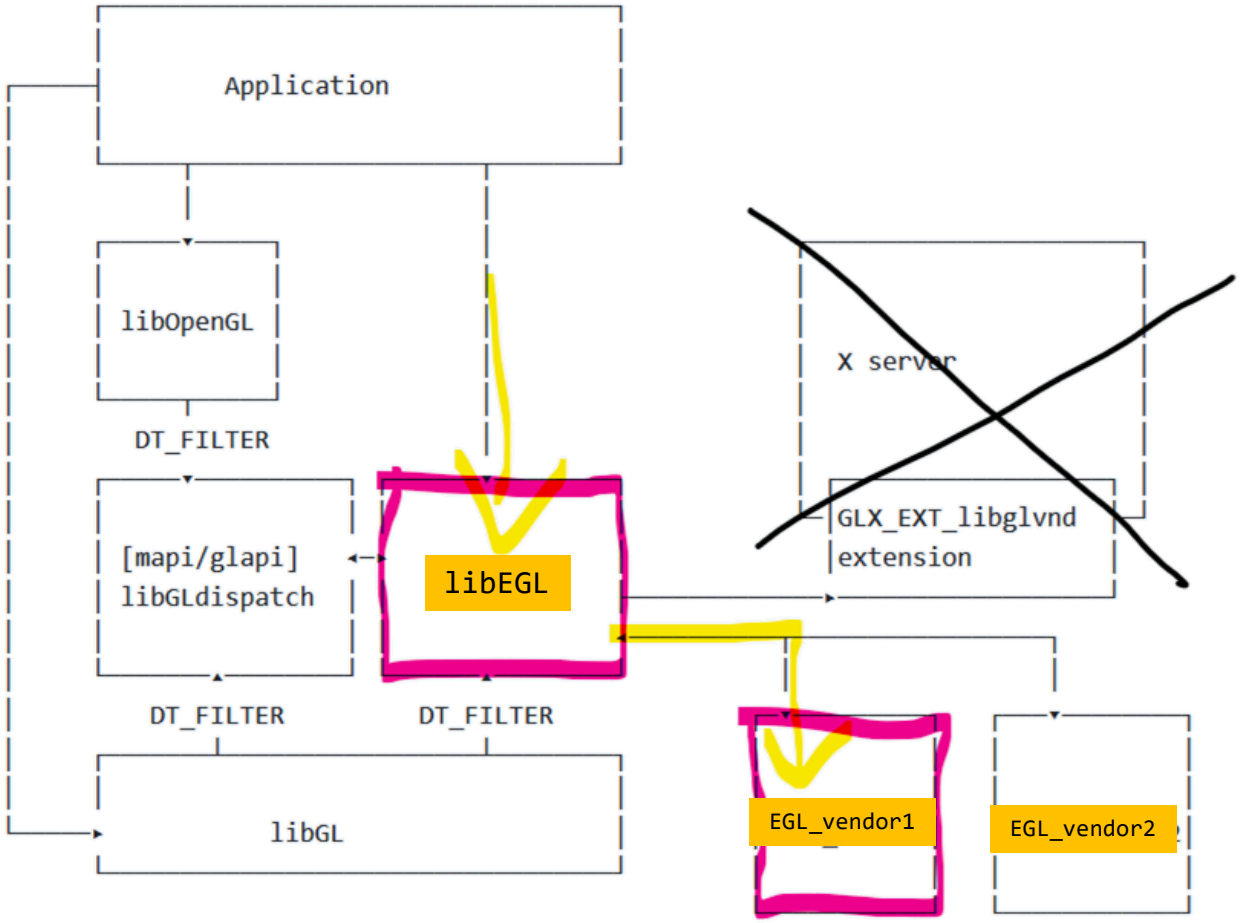
https://gitlab.freedesktop.org/glvnd/libglvnd

EGL dispatching

- For obvious reasons, EGL can't rely on asking an X server for a vendor name
- So, each vendor provides a JSON file (similar to how Vulkan ICD's are loaded)
- EGL will simply call into each vendor library until it finds one that succeeds.

EGL dispatching is based on vendor-provided JSON files

__EGL_VENDOR_LIBRARY_FILENAMES



DT_FILTER :
symbols exported by A
are resolved to entrypoints in B

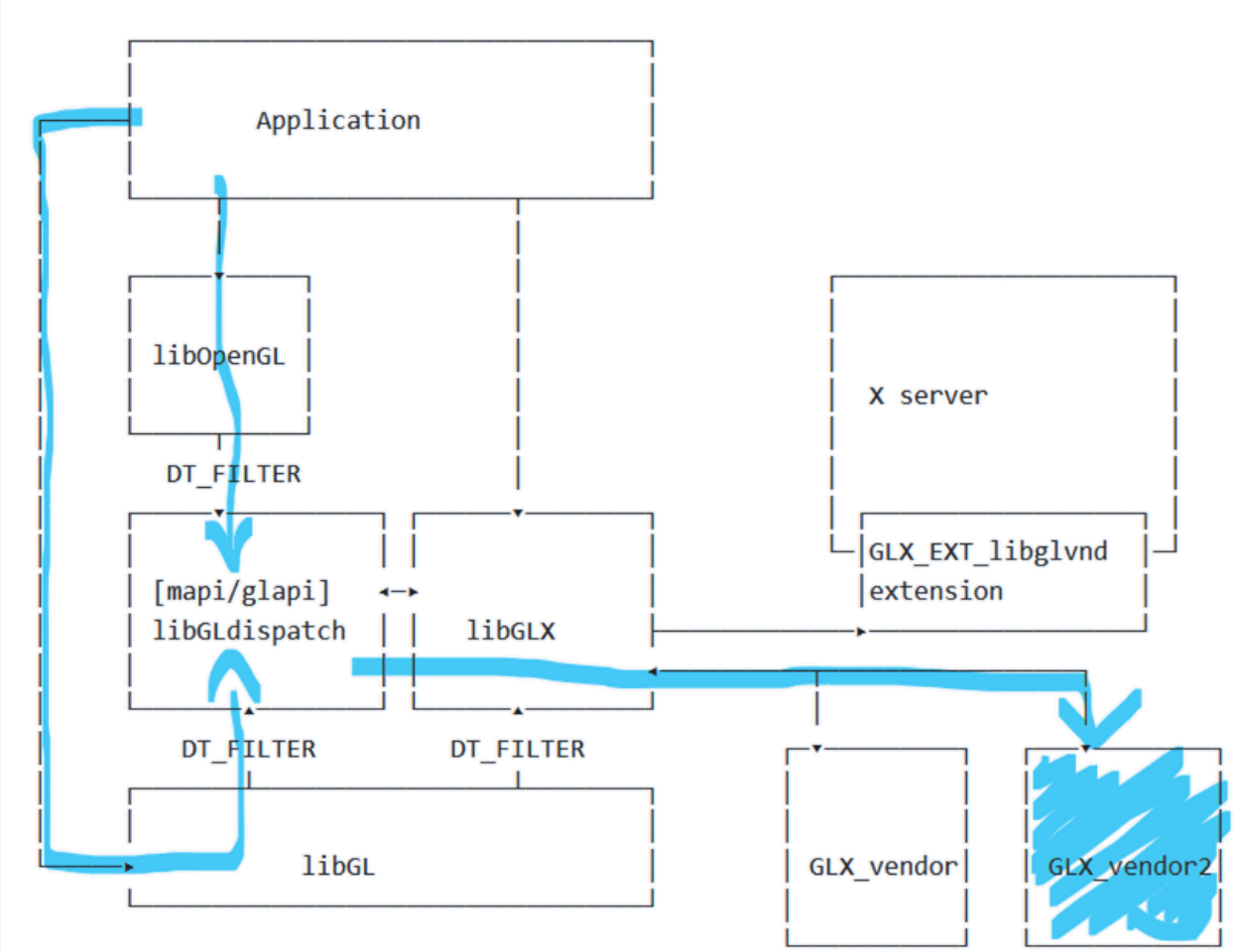
LIBGLVND (3)

Vendor-Neutral GL Dispatch Library

https://gitlab.freedesktop.org/glvnd/libglvnd

OpenGL dispatching

- All OpenGL functions are dispatched based on the current context.
- OpenGL dispatching is handled in libGLdispatch, which is used by both EGL and GLX.
- libGLdispatch uses a dispatch table to look up the correct vendor library.



Verify the active driver

- `glxinfo | grep "OpenGL vendor„`
- `eglinfo | grep "EGL_VENDOR“`

DT_FILTER :
symbols exported by A
are resolved to entrypoints in B

OPENGL MODULE AT JSC

A wrapper for MESA and more … used since 2020 @ JSC

```
easyblock = 'Bundle'
name = 'OpenGL'
version = '2024a'
components = [
        ('libglvnd', '1.7.0‘, {..}),
        ('Mesa', '24.2.8‘, {..}),
        ('glu', '9.0.3‘, {..}),
        ('glew', '2.2.0‘, {..}),
        ('demos', '9.0.0‘, {..}),
]
```

- The vendor neutral dispatch layer
- Mesa especially software rendering
- OpenGL Utility Library
  - offers higher level GL-graphics functions
- OpenGL Extension Wrangler Library
  - which OpenGL extensions are supported at run-time
- MESA demos
  - offers the important command 'eglinfo'

https://github.com/easybuilders/JSC/blob/2025/Golden_Repo/o/OpenGL/OpenGL-2024a-GCCcore-13.3.0.eb

JÜLICH Forschungszentrum

OPENGL MODULE AT JSC

A wrapper for MESA and more … used since 2020 @ JSC

```
easyblock = 'Bundle'
name = 'OpenGL'
version = '2024a'
components = [
        ('libglvnd', '1.7.0‘, {..}),
        ('Mesa', '24.2.8‘, {..}),
        ('glu', '9.0.3‘, {..}),
        ('glew', '2.2.0‘, {..}),
        ('demos', '9.0.0‘, {..}),
]
```

OpenGL Extension Wrangler Library
- which OpenGL extensions are supported at run-time

This is just GLEW for GLX and not GLEW for EGL
as GLEW selects GLX/EGL
at compile-time and not run-time
(https://github.com/nigels-com/glew/issues/172#issuecomment-357400019)

=> Extra module for GLEW-EGL needed
to enable GLEW for EGL.

https://github.com/easybuilders/JSC/blob/2025/Golden_Repo/o/OpenGL/OpenGL-2024a-GCCcore-13.3.0.eb

JÜLICH Forschungszentrum

OPENGL MODULE AT JSC

A wrapper for MESA and more … used since 2020 @ JSC

```
postinstallcmds = [
..
 '{ cat > %(installdir)s/share/glvnd/egl_vendor.d/10_nvidia.json; } << \'EOF\‘\n‘
..
 '{ cat > %(installdir)s/share/glvnd/egl_vendor.d/50_mesa.json; } << \'EOF\‘\n‘
..


modextravars = {
'__EGL_VENDOR_LIBRARY_FILENAMES‘: (
        '%(installdir)s/share/glvnd/egl_vendor.d/10_nvidia.json:‘
        '%(installdir)s/share/glvnd/egl_vendor.d/50_mesa.json‘
    ),
..
```

https://github.com/easybuilders/JSC/blob/2025/Golden_Repo/o/OpenGL/OpenGL-2024a-GCCcore-13.3.0.eb

JÜLICH Forschungszentrum

OPENGL MODULE AT JSC

A wrapper for MESA and more … used since 2020 @ JSC

```
%(installdir)s/share/glvnd/egl_vendor.d/10_nvidia.json
        {
          "file_format_version" : "1.0.0",
          "ICD" : {
             "library_path" : "libEGL_nvidia.so.0"
          }
        }
```

Needs to be in LD_LIBRARY_PATH
(incl. its dependencies)

```
%(installdir)s/share/glvnd/egl_vendor.d/50_mesa.json
        {
          "file_format_version" : "1.0.0",
          "ICD" : {
             "library_path" : " libEGL_mesa.so.0"
          }
        }
```

https://github.com/easybuilders/JSC/blob/2025/Golden_Repo/o/OpenGL/OpenGL-2024a-GCCcore-13.3.0.eb

JÜLICH Forschungszentrum

ADVANTAGES & LIMITATIONS

- libglvnd – The common OpenGL ABI for nowadays Linux distributions
- Allows to switch vendor‘s OpenGL library at runtime

  without the need of multiple modules for all OpenGL applications
- Possiblity to set the vendor on system side via environment variables

Open Questions

- Who ensures that the vendors libs mentioned in JSON are in LD_LIBRARY_PATH?
  - Should this be done by EasyBuild or on system level?
- GLX – Vendor needs to be set in X or forced with `__GLX_VENDOR_LIBRARY_NAME`
  - For automation, it needs some smart logic which sets this when the module gets loaded.
- EGL – We rely on the fact that EGL will call into each vendor library until it finds one that succeeds.
  - Is that rock-solid for other vendors (AMD-Catalyst)?

JÜLICH Forschungszentrum