Shading Languages Symposium 2026

The glslang Compiler: Present and Future

Arcady Goldmints-Orlov, LunarG

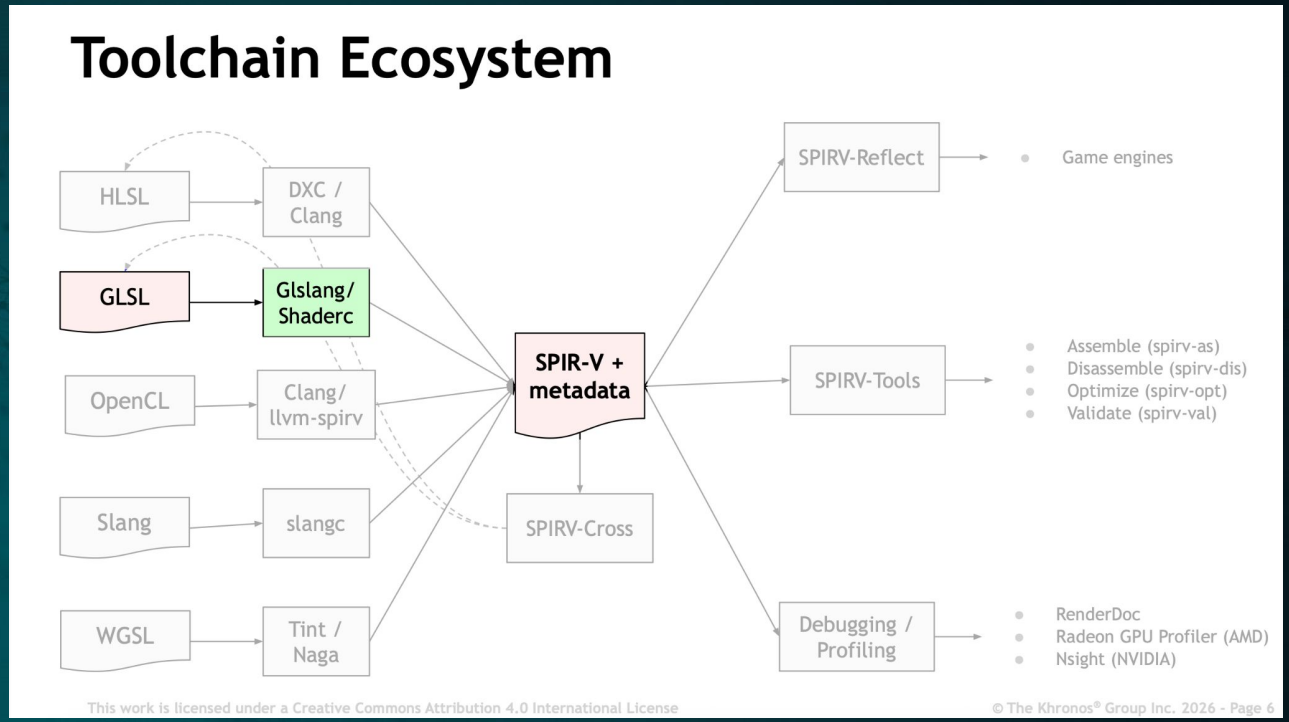
Some quick information about me

- Working at LunarG since 2023
- Started helping with glslang maintenance when I joined LunarG
- Created the KosmicKrisp NIR-to-MSL compiler
- Also on the V3D (Raspberry Pi 4) compiler in Mesa.
- Taught myself compilers on the job

This talk

- glslang is still alive
- We've been making some improvements, especially the API
- Where glslang is going in the future.
- How you can help

What we're talking about



LUNARG

A brief history of GLSL

- Originally a language built into OpenGL
- Implemented by each driver vendor in their driver
- Vendors created extensions, some extensions got incorporated into new versions of GLSL/OpenGL
- Vulkan introduces SPIR-V, language no longer in the driver.
- Main implementation is glslang, somewhat API agnostic
- No new versions have been released since 2017

A brief history of glslang

- Started out as a GLSL validator
  - Descended from a 3DLabs codebase from 2005
- When SPIR-V was created, it gained a SPIR-V backend
- Also an HLSL frontend (DXC didn't exist yet)
- Maintained by John Kessenich until he retired, then LunarG took over
- A fairly simple compiler, going from GLSL syntax tree to SPIR-V
  - SPIR-V was designed to make this easy
  - Optimization is in a separate tool entirely, spirv-opt

GLSL is still popular

- The ecosystem survey shows GLSL+glslang as the most popular option
  - Including with commercial users.
- Used in WebGL, Godot Engine, Blender
- Not just used by programmers
- Things like Shadertoy (www.shadertoy.com) provide an easy entry point for people to learn the language
- slang supports some GLSL but its support is far from complete
  - Doesn't support buffer_reference

glslang maintenance

- Done by LunarG with help from NVIDIA
- Merging new extensions
- Fixing bugs
  - Priority given to compiler crashes and issues in the GLSL frontend and SPIR-V backend
- Packaging improvements
- HLSL frontend is legacy
  - Incomplete implementation of Shader Model 5

Recent improvements: API/ABI

- glslang was not very well designed for use as a library
- Exposed all its guts in installed header files
  - This made it hard to make changes as any change could break compatibility for downstream projects.
- We cut down the installed headers to a reasonable subset providing a stable public API
- Limited exported symbols when building as a shared library
- We might have gotten some wrong, please file issues
- Bringing the C interface up to parity with the C++ one

Trying to be a better package citizen

- glslang was split into a bunch of different libraries
  - This made it difficult for projects using it as a dependency
  - And for distro packagers
- Now it's all in libglslang.so/libglslang.a
  - No more libGenericCodeGen libMachineIndependent libOSDependent etc.
  - Old libraries are kept as stubs for compatibility
- Working on shipping a pkg-config file

Removed spirv-remap

- This was a completely separate program from glslang
- Used for making SPIR-V binaries more “similar” for better compression of shader bundles
- It’s in spirv-tools now as the `--canonicalize-ids` option

Other recent improvements

- Lots of improvements to shader debuginfo thanks largely to NVIDIA contributions.
- Exposing SPIR-V features like GL_EXT_nontemporal_keyword
- Some new syntax extensions like GL_NV_explicit_typecast

Challenges

- GLSL core language development has stalled
  - New extensions to expose SPIR-V features, very little work to improve the language itself.
- Complex process to add new features, few non-IHV contributions
  - Need to write an extension spec first
  - Some counterexamples though, e.g. GL_EXT_nontemporal_keyword

Creating a new extension

- Write a spec, create a PR in the KhronosGroup/GLSL repo
  - Anyone can make a GL_EXT_whatever extension
- Write the extension scaffolding in glslang
  - There are good instructions in the Versions.cpp file
- Write the actual implementation for your new keywords/functions
  - Bison grammar in glslang.y
  - Grammar calls helper functions in ParseHelper.cpp
- Make a glslang PR
  - Make sure to add tests!

Ideas for the future

- Generics
  - GL_EXT_long_vector introduced a generic vector<T, N> syntax
- Separate compilation
  - You can compile “library” files using `--no-link`
  - Syntax support still needed for imports and exports
- A reference type?
- Other ideas? Submit issues, or better yet, write an extension spec!
  - Anyone can propose a GLSL extension spec
  - Once the spec is written, glslang can accept a PR to implement it

A bright future for GLSL?

- Should there be a new core version?
  - How would we make shadertoy support it?
- GLSL on Direct3D now that they will be using SPIR-V?
- Modern language features?
- SPIR-V does provide an opportunity to improve the language without needing to update multiple implementations first.

Conclusion

- glslang is still widely used
- The compiler is still moving forward
- Anyone can write an extension
- Let us know how we can make it better

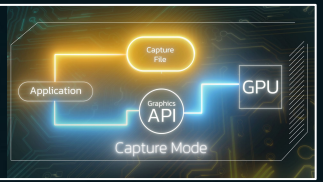
LUNARG
POWER YOUR SUCCESS

Thank you –
KHRONOS GROUP

KosmicKrisp

Come to the LunarG Table!

See KosmicKrisp & GFXReconstruct



Take the 2026 Vulkan Ecosystem Survey!

LunarG Presentations

Vulkanised 2026

LunarG Presentations

Shading Languages Symposium 2026


LUNARG
POWER YOUR SUCCESS