Nouveau/NVK Update

Faith Ekstrand

XDC 2025, Vienna Austria

About Me

- Faith Ekstrand
  - @gfxstrand@treehouse.systems
- Been around freedesktop.org since 2013
  - First commit: wayland/31511d0e, Jan 11, 2013
- At Intel from June 2014 to December 2022
  - NIR, Intel (ANV) Vulkan driver, SPIR-V → NIR, ISL, other Intel bits
- At Collabora since January 2022
  - Work across the upstream Linux graphics stack, wherever needed
  - Currently the lead developer / maintainer of NVK

COLLABORA

Okay, let’s talk about Nouveau

COLLABORA

Hardware enabling

Blackwell enabling

- Mostly Dave, Mohamed, Mel, and Faith

Blackwell enabling

- Mostly Dave, Mohamed, Mel, and Faith
- Getting hardware was a struggle
  - GPU availability early on was really bad
  - Dave got a GPU in March, Faith didn't get one until May
  - Mel never got one and just poked at the disassembler

Blackwell enabling

- Mostly Dave, Mohamed, Mel, and Faith
- Getting hardware was a struggle
- Headers took a while, too
  - To NVIDIA's credit, they did release them, just not immediately
  - The Compute QMD layout changed entirely
  - There were new bits added to the copy engine
  - 3D got split depth/stencil which meant new stencil packets

Blackwell enabling

- Mostly Dave, Mohamed, Mel, and Faith
- Getting hardware was a struggle
- Headers took a while, too
- Blackwell is biggest ISA change since Turing
  - Inline cbuf fetch is gone
  - Lots more things take UGPR extra sources
  - Lots more UGPR opcodes

Blackwell enabling

- Mostly Dave, Mohamed, Mel, and Faith
- Getting hardware was a struggle
- Headers took a while, too
- Blackwell is biggest ISA change since Turing
- But we got there!
  - Vulkan 1.4 conformant as of August 1

Kepler Enabling

- Mostly Lorenzo and Faith
  - Faith implemented the Kepler A encoding
  - Lorenzo implemented the Kepler B encoding and everything else

Kepler Enabling

- Mostly Lorenzo and Faith
- Most of the API side was already done for Maxwell

Kepler Enabling

- Mostly Lorenzo and Faith
- Most of the API side was already done for Maxwell
- Kepler has 2 instruction sets
  - Kepler A is the Fermi encoding with different texture/image ops
  - Kepler B is its own encoding

Kepler Enabling

- Mostly Lorenzo and Faith
- Most of the API side was already done for Maxwell
- Kepler has 2 instruction sets
- Kepler doesn't have surface ops
  - It has some magic ALU ops for clamping and doing tiling calculations
  - Load/store/atomic just takes a 40-bit address

Kepler Enabling

- Mostly Lorenzo and Faith
- Most of the API side was already done for Maxwell
- Kepler has 2 instruction sets
- Kepler doesn't have surface ops
- Kepler has weird shared atomics
  - They use a split load/lock, store/unlock like Arm CPUs

Kepler Enabling

- Mostly Lorenzo and Faith
- Most of the API side was already done for Maxwell
- Kepler has 2 instruction sets
- Kepler doesn't have surface ops
- Kepler has weird shared atomics
- Kepler can't do the Vulkan memory model
  - Required for Vulkan 1.3

Kepler Enabling

- Mostly Lorenzo and Faith
- Most of the API side was already done for Maxwell
- Kepler has 2 instruction sets
- Kepler doesn’t have surface ops
- Kepler has weird shared atomics
- Kepler can’t do the Vulkan memory model
- Vulkan 1.2 conformant as of June 29

COLLABORA

The switch to Zink

The Switch to Zink

The Switch to Zink

- Turing+ now uses Zink by default
  - Starting with Mesa 25.1
  - Users can still disable with `NOUVEAU_USE_ZINK=0`
  - Kepler through Volta will come later

The Switch to Zink

- Turing+ now uses Zink by default
- Blackwell is Zink-only
  - We did not enable the Nouveau GL driver for Blackwell
  - Given the in-depth compiler changes, probably a good call

The Switch to Zink

- Turing+ now uses Zink by default
- Blackwell is Zink-only
- We had to fix a lot of Zink bugs
  - Core rendering was solid, apart from a few issues we fixed early on
  - Most client apps were fine
  - Compositors, the X server, and web browsers were tricky
  - Mostly synchronization bugs
  - We still have a Firefox flicker bug

The Switch to Zink

- Turing+ now uses Zink by default
- Blackwell is Zink-only
- We had to fix a lot of Zink bugs
- Mesa 25.1 wasn’t totally smooth
  - There was a Kopper enabling bug with Nouveau
  - The path we use to force Zink didn’t enable Kopper
  - This forced Wayland users down broken legacy paths for
  - Fixed in Mesa 25.2 and we back-ported a hack to 25.1

The Switch to Zink

- Turing+ now uses Zink by default
- Blackwell is Zink-only
- We had to fix a lot of Zink bugs
- Mesa 25.1 wasn’t totally smooth
- Hopefully Mesa 25.2 will be smoother

The Switch to Zink

- Turing+ now uses Zink by default
- Blackwell is Zink-only
- We had to fix a lot of Zink bugs
- Mesa 25.1 wasn’t totally smooth
- Hopefully Mesa 25.2 will be smoother
- There are Nouveau kernel bugs NVK hits harder than NGL
  - This means Zink hits those bugs. 😿

The Switch to Zink

- Turing+ now uses Zink by default
- Blackwell is Zink-only
- We had to fix a lot of Zink bugs
- Mesa 25.1 wasn’t totally smooth
- Hopefully Mesa 25.2 will be smoother
- There are Nouveau kernel bugs NVK hits harder than NGL
- Firefox still has a flicker bug

How is Zink?

Not perfect but it’s working okay

COLLABORA

New features

COLLABORA

Vulkan 1.4

New features

- Vulkan 1.4
- VK_KHR_cooperative_matrix
- VK_EXT_depth_clamp_zero_one
- VK_KHR_fragment_shading_rate
- VK_KHR_global_priority
- VK_KHR_maintenance8
- VK_KHR_maintenance9
- VK_KHR_present_id2
- VK_KHR_present_wait2
- VK_KHR_shader_untyped_pointers
- VK_KHR_unified_image_layouts
- VK_EXT_hdr_metadata
- VK_EXT_zero_initialize_device_memory
- VK_AMD_buffer_marker
- VK_ANDROID_native_buffer
- VK_MESA_image_alignment_control
- descriptorBufferPushDescriptors
- shaderSharedInt64Atomics

COLLABORA

VK_KHR_cooperative_matrix

VK_KHR_cooperative_matrix

- Initially started by Mary almost a year ago
- Picked up by Karol and Dave
- After some optimization work, we're within 90% of proprietary driver perf for matrix multiply

COLLABORA

Performance improvements

COLLABORA

Instruction latencies and scheduling

Instruction latencies and Scheduling

- We have real latency information now!
  - Thanks, NVIDIA!

Instruction latencies and Scheduling

- We have real latency information now!
  - Turing+ have real numbers, not the R/E worst cases
  - Thanks, NVIDIA!
  - Incorporated by Dave

Instruction latencies and Scheduling

- We have real latency information now!
- Mel wrote us a post-RA scheduler
  - Runs after register allocation
  - Tries to reduce stalling and improve parallelism
  - Reduces static cycles by an estimated 14%

Instruction latencies and Scheduling

- We have real latency information now!
- Mel wrote us a post-RA scheduler
- Mel also wrote a pre-RA scheduler
  - Tries to reduce register pressure and increase occupancy
  - Mostly waiting on review

COLLABORA

Memory access improvements

Memory access improvements

- Nvidia ld/st instructions take GPR+UGPR+imm address

Memory access improvements

- Nvidia ld/st instructions take GPR+UGPR+imm address
- NIR doesn’t really do any of this
  - https://gitlab.freedesktop.org/mesa/mesa/-/merge_requests/33926
  - Even that’s not enough for Nvidia

Memory access improvements

- Nvidia ld/st instructions take GPR+UGPR+imm address
- NIR doesn’t really do any of this
- Karol has been working on improving it
  - Cooperative matrix benchmarks are a memory micro-benchmark
  - Some of it has landed
  - Some is still WIP

Memory access improvements

- Nvidia ld/st instructions take GPR+UGPR+imm address
- NIR doesn’t really do any of this
- Karol has been working on improving it
- We also need to improve bounds checking
  - This is currently destroying us in D3D titles
  - Faith’s predication MR improves it
  - Just predication isn’t enough
  - VK_NV_raw_access_chains?

COLLABORA

Image compression

Image compression

- Mohamed is working on this

Image compression

- Mohamed is working on this
- Currently in draft form

Image compression

- Mohamed is working on this
- Currently in draft form
- It requires Nouveau kernel driver changes

Image compression

- Mohamed is working on this
- Currently in draft form
- It requires Nouveau kernel driver changes
- It’s also tricky to enable in general
  - Compression is closely tied to alignments
  - Not particularly sparse friendly
  - Currently we rely on dedicated allocations

Image compression

- Mohamed is working on this
- Currently in draft form
- It requires Nouveau kernel driver changes
- It’s also tricky to enable in general
- As much as a 50% perf improvement in some games

COLLABORA

Other improvements

Other NVK perf improvements

- ZCULL (depth test acceleration)
  - Mel has been working on this for a bit
  - Also requires kernel changes
  - Currently waiting on review

Other NVK perf improvements

- ZCULL (depth test acceleration)
- Correct shared memory settings
  - We got shared memory (smem) settings wrong
  - They were good enough for correctness, bad for perf
  - We weren’t getting full occupancy
  - Fixing this was 2x speedup for cooperative matrix
  - Also helps compute-heavy games

Other NVK perf improvements

- ZCULL (depth test acceleration)
- Correct shared memory settings
- Compute MME on Ampere B+
  - Avoids unnecessary subchannel switching
  - A few more FPS in games

Other NVK perf improvements

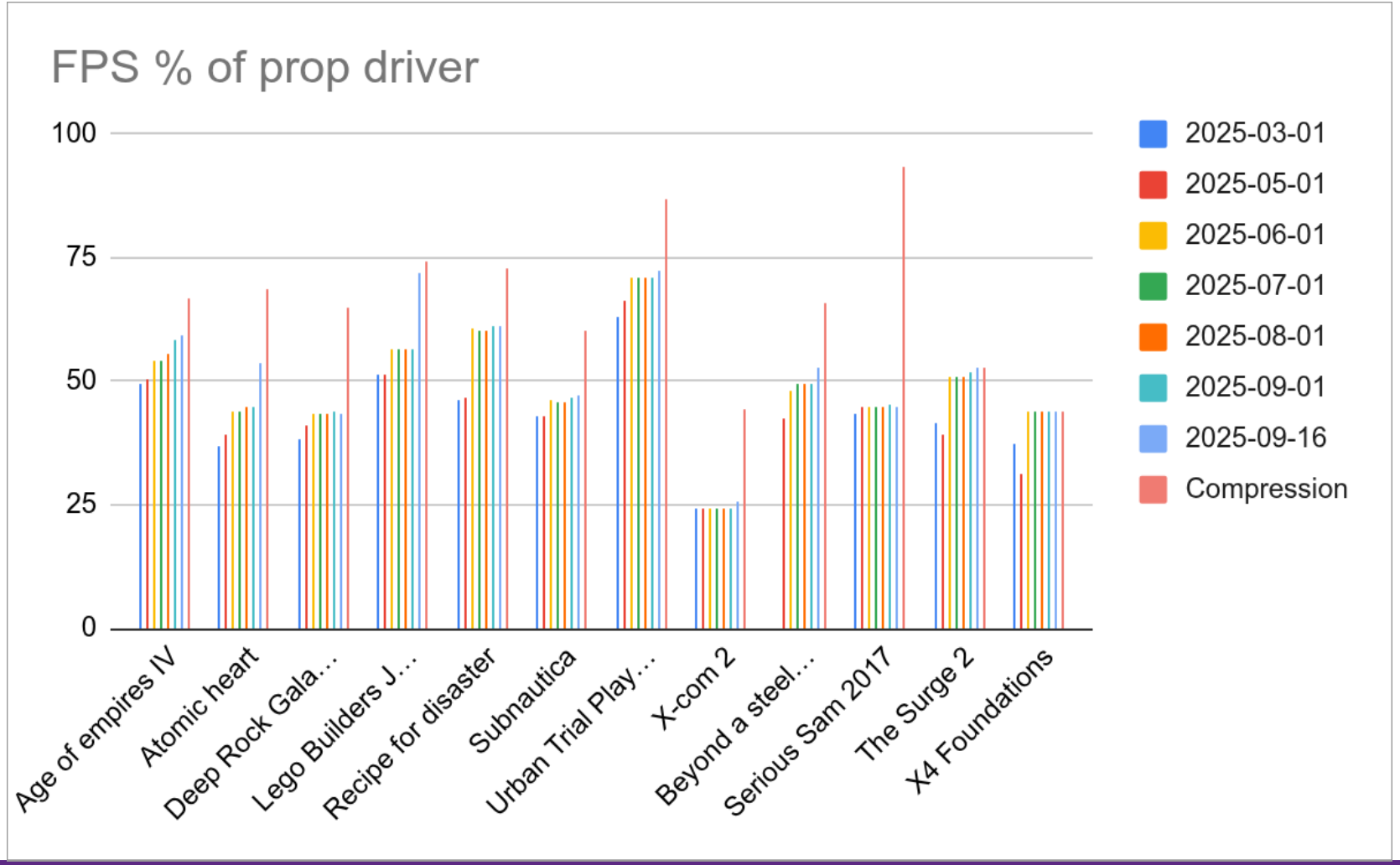
- ZCULL (depth test acceleration)
- Correct shared memory settings
- Compute MME on Ampere B+
- Reduce subchannel switching
  - Pre-Blackwell, subchannel switches cause stalls
  - Mel has a draft MR to reduce subchannel switching
  - Waiting on better Blackwell testing

Other NVK perf improvements

- ZCULL (depth test acceleration)
- Correct shared memory settings
- Compute MME on Ampere B+
- Reduce subchannel switching
- Dedicated transfer queues
  - Works but we ran into kernel synchronization bugs
  - Bugs are fixed but it takes time for the fix to propagate

Other NVK perf improvements

- ZCULL (depth test acceleration)
- Correct shared memory settings
- Compute MME on Ampere B+
- Reduce subchannel switching
- Dedicated transfer queues
  - Works but we ran into kernel synchronization bugs
  - Bugs are fixed but it takes time for the fix to propagate


FPS % of prop driver
100
75
50
25
0
2025-03-01
2025-05-01
2025-06-01
2025-07-01
2025-08-01
2025-09-01
2025-09-16
Compression
Age of empires IV
Atomic heart
Deep Rock Gala…
Lego Builders J…
Recipe for disaster
Subnautica
Urban Trial Play…
X-com 2
Beyond a steel…
Serious Sam 2017
The Surge 2
X4 Foundations

COLLABORA

Going forward

Going forward

- Vulkan video is in the works
  - Started by Daniel Almeida
  - Finally using Rust in driver code 🎉
  - Successfully decodes videos with fluster
  - Doesn’t pass CTS yet

Going forward

- Vulkan video is in the works
- We still need ray tracing
  - Konstantin did some R/E work on acceleration structures
  - Still don’t know how shaders work

Going forward

- Vulkan video is in the works
- We still need ray tracing
- We need more kernel developers
  - It's basically been Ben, Dave, and Danilo
  - None of them are full-time
  - We're excited for Nova but the Nova DRM driver is still slideware
  - We're barely keeping the lights on

Going forward

- Vulkan video is in the works
- We still need ray tracing
- We need more kernel developers
- Continue squashing Zink bugs
  - Zink is working well enough but there are still bugs

Going forward

- Vulkan video is in the works
- We still need ray tracing
- We need more kernel developers
- Continue squashing Zink bugs
- More performance!

We are hiring

col.la/careers