Nouveau/NVK Update

Faith Ekstrand
Karol Herbst

XDC 2023
Hopefully a little more interesting than your average “state of nouveau”...
Lots happening in nouveau these days!

**Kernel:**
- New nouveau kernel UAPI for bind/exec
- GSP firmware support
- New hardware support

**Userspace:**
- OpenGL improvements
- NVK improvements
- New shader compiler (NAK)
New nouveau UAPI

Driven by the needs of NVK:

- Arbitrary sized command buffers
- Memory aliasing of tiled images
- Sparse resources
- Explicit synchronization
New VM_BIND ioctl

- Explicitly binds ranges of BOs to virtual addresses
- Sparse binding fully supported
  - Page aliasing (binding the same page to multiple VA)
  - Sparse null pages for soft faults in shaders
  - Binds can be immediate or pipelined with in/out fences
- Tile mode and PTE kind are set in VM_BIND
  - Necessary for Vulkan image memory aliasing with tiled images
New EXEC ioctl

- Vastly simpler than the old one
- Assumes VM_BIND
- No BO list and no BO list walking in kernel space
  - All non-shared BOs share a single dma_resv
- Takes an array of (addr, len) pairs
- 100% explicit synchronization
Everything is drm_syncobj now

- No more implicit synchronization
  - Implicit sync is emulated with dma_buf_import/export_sync_file
- VM_BIND and EXEC both take arrays of syncobjs
  - One array for the waits and one array for the signals
- Timeline syncobjs are fully supported
GSP Firmware

- Slowly getting merged
- Requires 25MiB+ firmware blobs
  - Distributions have to figure out their initramfs situation
  - /boot/ partitions can be too small to support this
  - initramfs generators should cull unused firmware files
  - Dave working on the kernel and initramfs support
- Follows Nvidia driver versioning/releases
- We use the exact same files Nvidia uses in their driver
Ampere and Ada support

- Ada Requires GSP
- OpenGL support in 23.0 (Ampere) and 23.2 (Ada)
- Should just work in NVK
Hopper support

- Compute GPU with very slow 3D support
- No plans to support it due to lack of hardware.
OpenGL improvements

- Multi-threaded OpenGL fixed in Mesa 22.3
- nv30 now runs modern gnome with Mesa 23.1
Since XDC 2022...
We implemented a few features...

- geometryShader
- tessellationShader
- shaderImageGatherExtended
- shaderStorageImageReadWithoutFormat
- sparseBinding
- sparseResidencyBuffer
- VK_KHR_bind_memory2
- VK_KHR_buffer_device_address
- VK_KHR_external_fence
- VK_KHR_external_fence_capabilities
- VK_KHR_external_fence_fd
- VK_KHR_external_memory
- VK_KHR_external_memory_capabilities
- VK_KHR_external_memory_fd
- VK_KHR_external_semaphore
- VK_KHR_external_semaphore_capabilities
- VK_KHR_external_semaphore_fd
- VK_KHR_depth_stencil_resolve
- VK_KHR_device_group
- VK_KHR_draw_indirect_count
- VK_KHR_driver_properties
- VK_KHR_dynamic_rendering
- VK_KHR_maintenance2
- VK_KHR_maintenance3
- VK_KHR_maintenance4
- VK_KHR_map_memory2
- VK_KHR_multiview
- VK_KHR_relaxed_block_layout
- VK_KHR_sampler_ycbcr
- VK_KHR_shader_clock
- VK_KHR_shader_draw_parameters
- VK_KHR_shader_viewport_index_layer
- VK_KHR_spirv_1_4
- VK_KHR_uniform_buffer_standard_layout
- VK_KHR_variable_pointers
- VK_KHR_variable_pointers
- VK_KHR_variable_pointers
- VK_EXT_4444_formats
- VK_EXT_buffer_device_address
- VK_EXT_conditional_rendering
- VK_EXT_descriptor_indexing
- VK_EXT_depth_clip_control
- VK_EXT_depth_clip_enable
- VK_EXT_extended_dynamic_state
- VK_EXT_extended_dynamic_state2
- VK_EXT_extended_dynamic_state3
- VK_EXT_external_memory_dma_buf
- VK_EXT_image_2d_view_of_3d
- VK_EXT_image_robustness
- VK_EXT_image_view_min_lod
- VK_EXT_index_type_uint8
- VK_EXT_line_rasterization
- VK_EXT_mutable_descriptor_type
- VK_EXT_non_seamless_cube_map
- VK_EXT_pci_bus_info
- VK_EXT_physical_device_drm
- VK_EXT_provoking_vertex
- VK_EXT_robustness2
- VK_EXT_sample_locations
- VK_EXT_sampler_filter_minmax
- VK_EXT_separate_stencil_usage
- VK_EXT_shader_demote_to_helper_invocation
- VK_EXT_shader_viewport_index_layer
- VK_EXT_transform_feedback
- VK_EXT_vertex_attribute_divisor
- VK_KHR_map_memory2
- VK_KHR_multiview
- VK_KHR_relaxed_block_layout
- VK_KHR_sampler_ycbcr
- VK_KHR_shader_clock
- VK_KHR_shader_draw_parameters
- VK_KHR_spirv_1_4
- VK_KHR_uniform_buffer_standard_layout
- VK_KHR_variable_pointers
- VK_KHR_variable_pointers
- VK_KHR_variable_pointers
- VK_EXT_4444_formats
- VK_EXT_buffer_device_address
- VK_EXT_conditional_rendering
- VK_EXT_descriptor_indexing
- VK_EXT_depth_clip_control
- VK_EXT_depth_clip_enable
- VK_EXT_extended_dynamic_state
- VK_EXT_extended_dynamic_state2
- VK_EXT_extended_dynamic_state3
- VK_EXT_external_memory_dma_buf
- VK_EXT_image_2d_view_of_3d
- VK_EXT_image_robustness
- VK_EXT_image_view_min_lod
- VK_EXT_index_type_uint8
- VK_EXT_line_rasterization
- VK_EXT_mutable_descriptor_type
- VK_EXT_non_seamless_cube_map
- VK_EXT_pci_bus_info
- VK_EXT_physical_device_drm
- VK_EXT_provoking_vertex
- VK_EXT_robustness2
- VK_EXT_sample_locations
- VK_EXT_sampler_filter_minmax
- VK_EXT_separate_stencil_usage
- VK_EXT_shader_demote_to_helper_invocation
- VK_EXT_shader_viewport_index_layer
- VK_EXT_transform_feedback
- VK_EXT_vertex_attribute_divisor
We merged NVK into mesa/main

nvk: A new Vulkan driver for Nvidia hardware

Merged Faith Ekstrand requested to merge nouveau/mesa:NVK/main into main 2 months ago

Overview 69 Commits 874 Pipelines 67 Changes 132

NVK: A new Vulkan driver for Nvidia hardware

This MR adds a new Vulkan driver for Nvidia hardware. This has been in the works since last May if the git logs are to be believed. For those interested in the development progress of the driver and a bit of why we think now is the time to make NVK happen, I've written a couple blog posts on the topic:
We merged NVK into mesa/main

- Build with -Dvulkan-drivers=nouveau-experimental
- Supports Turing (RTX 20XX and GTX 16XX) and later GPUs
  - Kepler+ will enumerate with NVK_I_WANT_A_BROKEN_VULKAN_DRIVER=1
- Requires a Linux 6.6 kernel
  - The new UAPI is a hard requirement
- Fails 4 Vulkan CTS tests
Oh, and I wrote a new compiler...

Draft: nouveau, nvk: Add a new back-end compiler for NVIDIA hardware

Faith Ekstrand requested to merge `gfxstrand/mesa:nak/main` into `main` 1 month ago

What does this MR do and why?

This adds the new back-end compiler for NVK that I've been cooking for about 6 months now. It's written in Rust, SSA all the way through, and designed assuming competent NIR input. The long-term plan is that this will be the compiler for NVK. We're a ways from that goal but far enough along that I feel pretty okay posting it as a draft MR. We may use it for GL one day but there's a decent chance that'll just be Zink.
Oh, and I wrote a new compiler...

• Brand new codebase
• Written in Rust
• Currently targeting Turing
  - Ampere kinda works
  - Mary has a Maxwell branch
• More on that in the next talk....
And we’re passing a LOT of CTS tests...

Test run totals:

- Passed: 401810/3988868 (10.1%)
- Failed: 0/3988868 (0.0%)
- Not supported: 3587054/3988868 (89.9%)
- Warnings: 4/3988868 (0.0%)
- Waived: 0/3988868 (0.0%)
And we’re passing a LOT of CTS tests...

Test run totals:
- Passed: 401810/3988868 (10.1%)
- Failed: 0/3988868 (0.0%)
- Not supported: 3587054/3988868 (89.9%)
- Warnings: 4/3988868 (0.0%)
- Waived: 0/3988868 (0.0%)
Does NVK run <X> yet?

I don’t know. 🤷🏻‍♀️ Maybe? Feel free to give it a try!

Over-all, I’ve been more focused on trying to get the core of the driver solid and bring up the new compiler than worrying about app compatibility. We’ll get to apps, just maybe not this year.
What Vulkan version does NVK advertise?

Currently, we advertise Vulkan 1.0

We have enough we probably could advertise Vulkan 1.1 and be non-conformant without breaking too much stuff but I wanted to get us across the 1.0 line first.
When will we get Vulkan 1.3?

Soon. It’s mostly compiler work at this point. That perfect CTS pass rate was with the new compiler and there are still a few bits missing.

Subgroups are the big blocker. There’s also some control-flow stuff and maybe a few other things.
Community updates
Unfortunately.....

Hi all,

As you may have gathered from the MAINTAINERS patch I just sent out, I have resigned from my position at Red Hat, and will be stepping back from nouveau development.

This is a personal decision that I've been mulling over for a number of years now, and I feel that with GSP-RM greatly simplifying support of future HW, and the community being built around NVK, that things are in good hands and this is the right time for me to take some time away to explore other avenues.

I still have a personal system with an RTX 4070, which I've been using the nouveau GSP-RM code on for the past couple of weeks, so chances are I'll be poking my nose in every so often :)

I wish everyone the best, and look forward to seeing the progress you all make on nouveau in the future.

Happy hacking!

Ben.
Community updates

- Ben stepped down as maintainer
- Lyude and Danilo will cover the kernel bits
- Faith is maintaining the new NVK driver
- A lot of new contributors to NVK
- Karol focuses on maintaining the GL driver
- Interested in helping? Please reach out!
Q: “Since it’s now possible to have re-clocking support, I wonder if the #nouveau team is interested in making an open-source reimplementation of CUDA?”

A: It’s complicated....
Users were asking...

Q: “Should my next GPU be NVIDIA?”

A: Not unless you want to help out. 😅

Maybe in another year or two...
Thank you!
Got Questions?