Having fun with GPU resets :D

André Almeida
XDC 2023
Hi!

- Kernel developer
- Working in the Steam Deck
Oh no, my GPU hanged!

- You are playing your game on Linux
- Something wrong is sent to the device
- ???
- Game over, reboot your machine
Modern GPUs are complex

- *Really* complex
- AMD Radeon RX 7900 XTX
  - 96 Compute units, 384 texture units, 6 shader engines, 58 B transistors...
- Shaders are Turing Complete
Modern GPUs are complex
Modern GPUs are complex

- If you have an infinity loop in the CPU, it's not that bad
- CPU programs has virtual memory and virtual processor
  - Things might be more barebone in the GPU
- But in a GPU, the display won't be able to update
Detecting GPU hangs
From the hardware to the application
Detecting GPU hangs

Device to Kernel

- Submit a job and wait until is done (halting problem)
  - Check fences
  - Or timeouts
- The kernel driver does a GPU reset
  - This can be "soft" resets, one hw engine/context reset or full device reset
    - Discrete amdgpu struggles with per-context resets
  - More complete resets are more destructive
  - Total lost of VRAM
- Now, report to userspace
Reporting GPU hangs

Kernel to Mesa

- DRM has no API for that
  - I915_GET_RESET_STATS
  - AMDGPU_CTX_OP_QUERY_STATE2
  - MSM_PARAM_FAULTS
  - Return -ERROR for ioctls
  - Nothing for Raspberry or Vivante (?)

- It's not really hw specific
  - But there's no common DRM context concept
Reporting GPU hangs

Mesa to application

Vulkan

- Before submitting commands or wait operation, Mesa asks the kernel if the device is around
- Otherwise, it propagates `VK_ERROR_DEVICE_LOST` to the app
- Then the app (maybe) do something about it
  - Recreate the context
  - Or just exit
- `VK_EXT_device_fault` to query info about the reset
Reporting GPU hangs

Mesa to application
OpenGL

- Before context creation and cs flush do a reset check
- If the app has `GL_ARB_robustness` support, it propagates an error for the app
- Otherwise:
  - Some drivers just kill the application
  - Other just block new calls
Reporting GPU hangs

Mesa to application

- APIs provide a way to tell apps that a reset happened:
  - `VK_ERROR_DEVICE_LOST`
  - `GL_ARB_robustness`
  - Non-robust GL apps are just killed
- But even robust apps can misbehave
- Mesa/kernel can ban fd if they keep resetting the GPU
Reporting GPU hangs

Mesa to application

• Applications then can recreate the context
• Usually that means:
  ◦ Oh no, my submit command returned a reset error
    ■ The GPU state is corrupted, but the CPU still good!
  ◦ Let's recreate all contexts, buffers, shaders, etc
  ◦ Cool, let's go!
• Bad apps may fall in a broken loop
#version 330 core

out vec4 FragColor;

void main()
{
    for (float x = 0.01; x < 1; x--) {
        FragColor = vec4(x * 1.0f, 1.0f, 1.0f, 1.0f)
    }
}
What can we do here?

- DRM <-> Mesa it's not really hw specific
  - How about we have a `DRM_GET_RESET_STATE`?
- `drm/doc`: Document DRM device reset expectations: A DRM documentation explaining what DRM drivers and Mesa should do when a reset happens
struct drm_get_reset {
    /** Context ID to query resets (in) */
    __u32 ctx_id; // no global context ID...

    /** Flags (out) */
    __u32 flags;

    /** Global reset counter for this card (out) */
    __u64 reset_count;

    /** Reset counter for this context (out) */
    __u64 reset_count_ctx;
};
drmGetReset(ctx_id, &reset);

if (reset.reset_count_ctx)
    return PIPE_GUILTY_CONTEXT;
if (reset.reset_count)
    return PIPE_INNOCENT_CONTEXT;
What happens in practice?

- There was no reset check at RADV

```
+ device->vk.check_status = radv_check_status;
```
What happens in practice?

• For RadeonSI, there was reset check...
• But the driver would return `GL_CONTEXT_RESET_ARB` forever
• So the app would recreate the context, get a reset return and recreate the context, get a reset return, ad infinitum
What happens in practice?

- For Iris (Intel), there is reset check...
- But it only notify the guilty application that a reset happened
- The guilty application then quits/recreate the context
- The rest aren't notified and counts on being luck to still be alive
  - Some resets are more destructive than others
What happens in practice?

Not much shared code, standardization, tests, validation...
What happens in practice?

- Each vendor reacts differently to resets
  - My focus is on amdgpu
- The state for discrete is that it would be unrecoverable for any kind of reset
- Just a black screen and not responsive. Access via ssh/tty sometimes worked
- Pierre-Eric (AMD) and improved this for KDE compositor
  - radeonsi wasn't following the spec
  - More testing is needed for robustness
What happens in practice?

• Other OSs have more control in the stack, so they can be more reliable
  ○ In particular in the compositor side, so it's easier to get in a standard behavior
Good reporting of GPU hangs

- Apart from reporting to userspace that the GPU was reset, would be nice to tell what happened
- Currently Mesa developers have a hard time figuring out what in the game caused the hang
Good reporting of GPU hangs

- GPU hang have two main sources:
  - Hardware settings (voltage, frequency)
  - Application errors (infinite loops)
  - There's no way to distinguish this right now
Good reporting of GPU hangs

- Ideally without overhead so can be enabled by default
- I proposed `AMDGPU_INFO_GUILTY_APP` to capture data about the hanged app (e.g. buffer in use)
  - This callbacks need to be platform specific
  - Reads some registers about which buffer was in use
  - AMD replied that we can't trust register values after a reset
  - We probably need some firmware support at this
Good reporting of GPU hangs

- **RADV_DEBUG=hang** isn't always effective, it changes the ordering of jobs
- Challenge: when the GPU hangs the hardware state can be a bit unreliable.
  - How to get the right info correctly?
  - Using the GPU in "debug" mode or inserting fences, barrier and extra information causes overhead
  - No easy way to deploy to all users
Roadmap for better GPU resets

- Standardization of how DRM reports GPU hangs to userspace
  - of how usermode driver deals with a hang and with the guilty application
  - what compositors should do after a hang
- Better hang log
  - Show which buffer caused the hang
  - Dump hardware state reliably
  - devcoredump
- Tests!
Links

https://lore.kernel.org/lkml/20230501185747.33519-1-andrealmeid@igalia.com/
https://lore.kernel.org/lkml/20230424014324.218531-1-andrealmeid@igalia.com/
https://lore.kernel.org/dri-devel/20230929092509.42042-1-andrealmeid@igalia.com/
https://gitlab.freedesktop.org/mesa/mesa/-/merge_requests/22290
https://gitlab.freedesktop.org/mesa/mesa/-/merge_requests/22253
Thanks!

igalia.com/jobs