Putting the CL in VulCLan

Alyssa Rosenzweig

vtn_bindgen2

1. Write library functions
2. Call them from your NIR pass
3. That’s it!

Self-contained nir_builder routines = no linking step.

https://gitlab.freedesktop.org/mesa/mesa/-/merge_requests/33099

nir_precompiled

1. Write OpenCL kernel taking kernel args
2. “Call” the kernel from your driver, passing args directly + dispatch info
3. That’s it!

Automatic data layouts via reflection = no boilerplate

https://gitlab.freedesktop.org/mesa/mesa/-/merge_requests/32339

libcl - our standard library

Printf, assert, mesa's src/util, and more.

Easy code sharing between host & device.

https://gitlab.freedesktop.org/mesa/mesa/-/merge_requests/32529

Host (dispatch)

```
increment_cs_invocations(cs, agx_1d(1), grid, stat, local_size);
```

Device

```
KERNEL(1)
increment_cs_invocations(global uint *grid, global uint32_t *statistic,
                         uint32_t local_size_threads)
{
   assert(local_size_threads >= 1 && local_size_threads <= 32);
   *statistic += local_size_threads * grid[0] * grid[1] * grid[2];
}
```

Output

```
Shader assertion fail at src/asahi/libagx/query.cl:107
Expected local_size_threads >= 1 && local_size_threads <= 32
```