EBC - A new backend compiler for etnaviv

Christian Gmeiner

2024-10-09

Agenda

1. Motivation
2. Status Quo
3. Target ISA
4. Road to Success
5. Shader Debugging Techniques
6. Challenges
7. Further Work

Motivation

Motivation

- Become a grownup compiler.
- Make it easier to enable new compiler based features.
- Wider use of Rust in Mesa.

Status Quo

Status Quo

- There are many good resourced on mesa compilers.
- PoC backend compiler based on agx.
- etnaviv gained a Rust based assembler (lib).
- NAK has done all the hard work.
- Shared Rust code for compilers.

Target ISA

Target ISA

- 128 bit fixed size ISA.
- up to 3 srcs.
- vec4.
- 2 types of registers (temps, uniforms).
- has isaspec support (etnaviv.xml).

Target ISA

```
imadlo0.u32       t0.x___, t0.xxxx, 32, u0.zzzz
```

Road to Success

Road to Success

- Do as much lowering in NIR as practical.
- Have a basic set of backend optimizers.
- Support control flow from day 0.
- Support all HW bit sizes (8/16/32) from day 0.

Road to Success

- Started about 3-4 months ago.
- Trying to build it right this time.
- Thinking long-term and focused on getting the architecture right.

Road to Success

A brief overview of the past few weeks (1):

- Keep it a secret as long as you can.
- Hack on a Copy & Paste thing for some weeks.
- Refactor everything many many times.
- The first OpenCL shaders are working.
- Start to work through piglit tests.
- Comparing EBC shaders with blob ones.

Road to Success

A brief overview of the past few weeks (2):

- Still refactorings and bug fixes.
- Passing 200-300 piglit tests.
- Started talking about my PoC with Faith.
- src/compiler/rust is born.
- Started talking about OpenCL topics with Karol.
- Still unsure about the PoC.

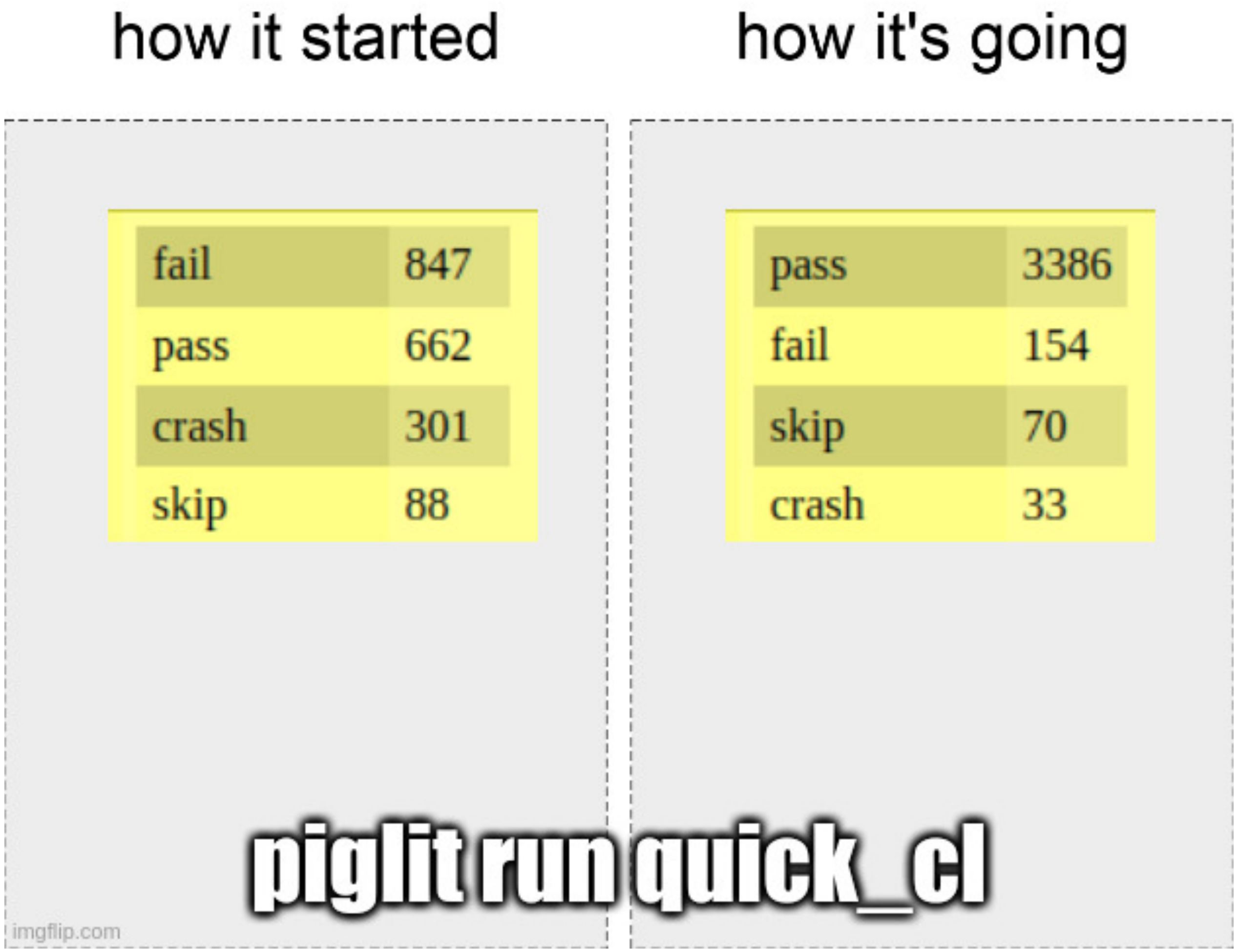
Road to Success

A brief overview of the past few weeks (3):

- Slowly passing around 1000 piglits.
- Frustration and lot of shader debugging.
- More bug fixes and unit tests.
- 2500 passing piglits.
- Liveness and RA added.
- 3000 passing piglits.

igalia

Road to Success



how it started

fail	847
pass	662
crash	301
skip	88

how it's going

pass	3386
fail	154
skip	70
crash	33

piglit run quick_cl

imgflip.com

Shader Debugging Techniques

Shader Debugging Techniques

The more tests where passing the bigger and complexer shaders got.

Shader Debugging Techniques

"See" the flow of data in the shader.

Shader Debugging Techniques

```
/*!
[test]
name: sqrt float1
kernel_name: sqrt
global_size: 1 0 0

arg_out: 0 buffer float[1] 2.0 tolerance 3 ulp
arg_in: 1 buffer float[1] 4.0
!*/

kernel void sqrt(global float* out, global float* in0)
{
  out[get_global_id(0)] = native_sqrt(in0[get_global_id(0)]);
}
```

Shader Debugging Techniques

```
000 load.denorm.u32.ls2  t2.x___, u0.yxxx, t0.xxxx, void
001 sqrt                 t3.x___, void, void, t2.xxxx
002 store.denorm.u32.ls2 mem.x___, u0.xxxx, t0.xxxx, t3.xxxx
003 store.denorm.u32     mem.x___, u0.zxxx, 0, t0.xxxx
004 store.denorm.u32     mem.x___, u0.zxxx, 4, t0.yxxx
005 store.denorm.u32     mem.x___, u0.zxxx, 8, t0.zxxx
006 store.denorm.u32     mem.x___, u0.zxxx, 12, t0.wxxx
...
018 store.denorm.u32     mem.x___, u0.zxxx, 60, t3.wxxx
```

Shader Debugging Techniques

```
arg_out: 0 buffer float[1] 2.0 tolerance 3 ulp
arg_in: 1 buffer float[1] 4.0
```

```
000 load.denorm.u32.ls2  t2.x___, u0.yxxx, t0.xxxx, void
001 sqrt                 t3.x___, void, void, t2.xxxx
```

```
t2.x: 40800000 - 1082130432 - 4.000000
t2.y: 00000000 - 0 - 0.000000
t2.z: 00000000 - 0 - 0.000000
t2.w: 00000000 - 0 - 0.000000
t3.x: 40000000 - 1073741824 - 2.000000
t3.y: 00000000 - 0 - 0.000000
t3.z: 00000000 - 0 - 0.000000
t3.w: 00000000 - 0 - 0.000000
```

Challenges

Challenges

- Having two compilers for one driver
- Untyped NIR
- Still lot of RE needed

Further Work

Further Work

Integer Promotions

Further Work

```
kernel void add(global char* out, char a, char b)
{
        out[0] = a + b;
}
```

```
8     %3 = @load_kernel_input (..)
8     %5 = @load_kernel_input (..)
32    %7 = i2i32 %5
32    %6 = i2i32 %3
32    %8 = iadd.nsw %6, %7
8     %9 = u2u8 %8
```

Further Work

- Expand the usage of the compiler to more shader stages.
- Expand the compiler to support more GPU generations.
- Fully replace the old compiler.
- Instruction stress tester.

Discussion

Join us!

https://www.igalia.com/jobs

igalia