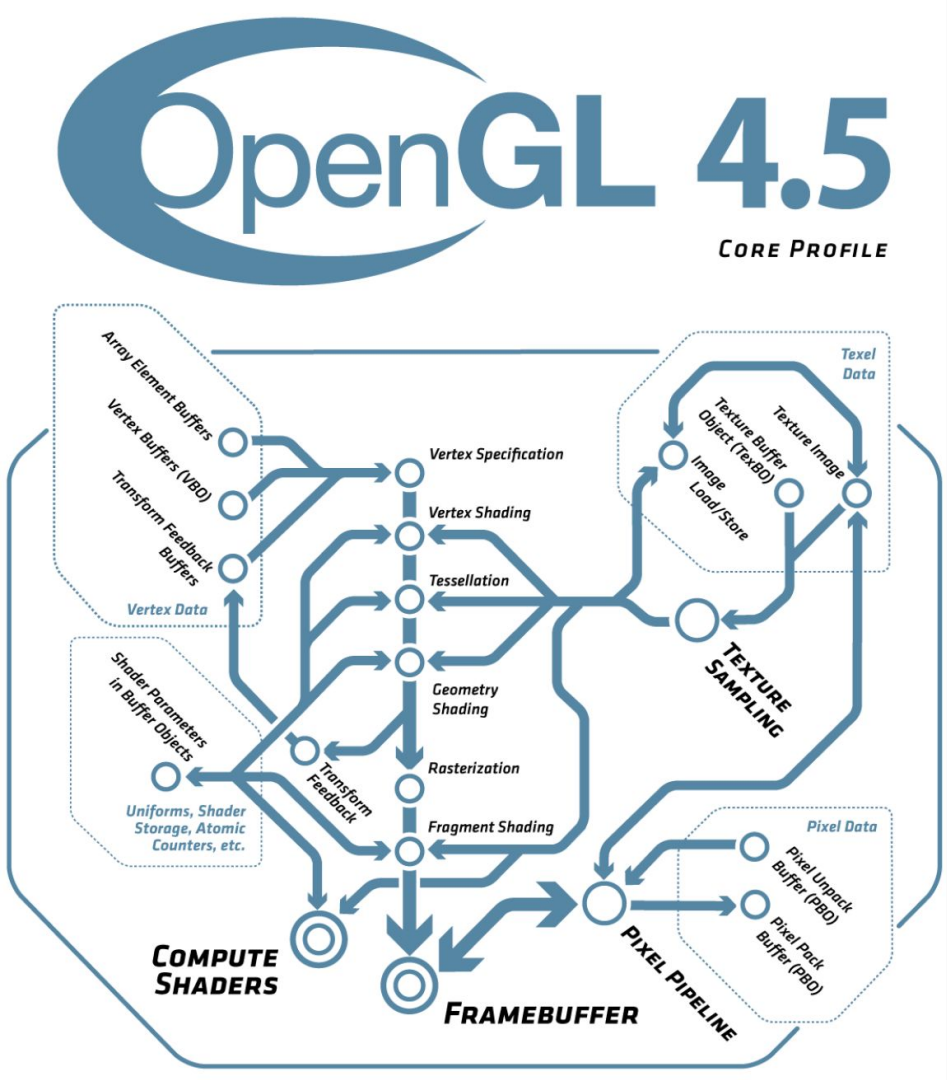
Blis

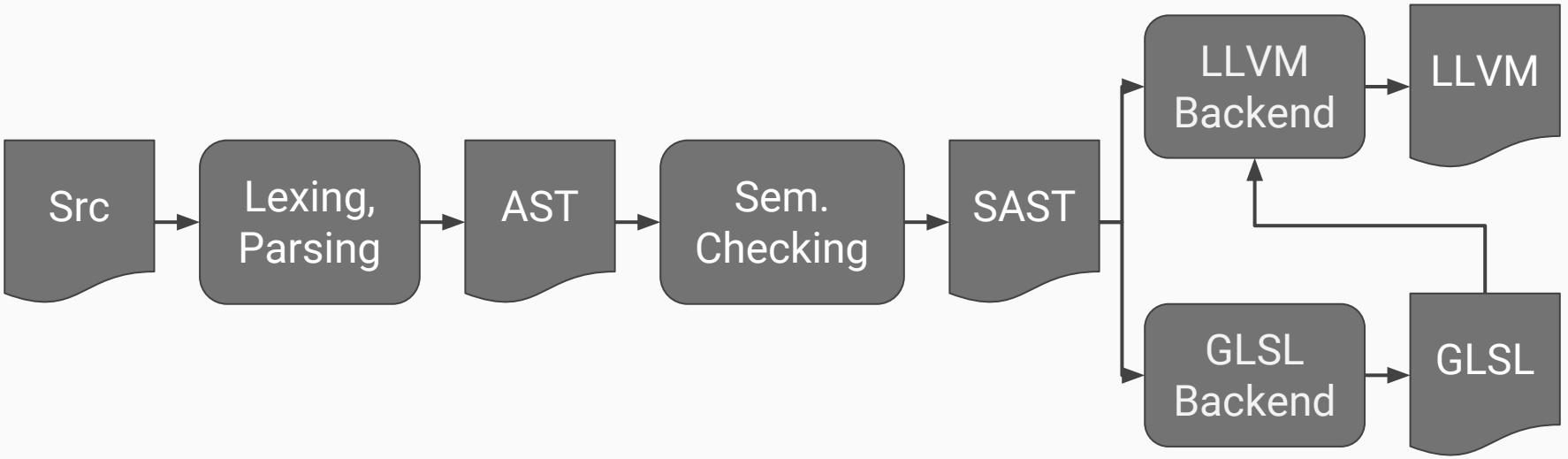
Connor Abbott, Wendy Pan, Klint Qinami, Jason Vaccaro

Motivation: Why Blis?

OpenGL is Complicated



Architecture



Comparing OpenGL and Blis Code

OpenGL

```
glBindBuffer(GL_ARRAY_BUFFER, pos_bo);
glEnableVertexAttribArray(pos_attr_loc);
glVertexAttribPointer(
     pos_attr_loc,
     4, GL_FLOAT, GL_FALSE,
     sizeof(point4),
     (void *) 0);
```

Blis

```
p.pos = b;
```

Features (Arrays, Matrices, Structs)

```
int[2][3] a = int[2][3](int[3](1, 2, 3), int[3](4, 5, 6));
int[] b = int[](2);

vec2 c = vec2(1337., 42.);
vec2 d = vec2(1., 2.);
vec2 e = vec2(4., 5.);

mat3x2 f = mat3x2(c, d, e);

u8vec4 g = u8vec4('a', 'b', 'c', 'd');
```

Features (cont.)

```
// Coef wise
ivec2 a = ivec2(1337, 42);
ivec2 b = ivec2(2, 2);
ivec2 c = b * a - a;
ivec2 d = a / b;

bvec2 e = bvec2(true, false);
e = !e;
```

Features (cont.)

```
mat3x3 A = mat3x3(vec3(7., 0., -3.), vec3(2., 3., 4.), vec3(1., -1.,
-2.));
mat3x3 Ainv = mat3x3(vec3(-2., 3., 9.), vec3(8., -11., -34.), vec3(-5.,
7., 21.));
mat3x3 I = A * Ainv;
mat3x3 I2 = Ainv * A;

// I and I2 are now identity matrix
```

Features (cont.)

```
struct foo {
    int a;
    float b;
};
int main() {
    struct foo temp = struct foo(42, 1337.0);
    printi(temp.a);
    printf(temp.b);
    return 0;
}
```

Features (cont.)

```
@vertex vec4 vshader(vec3 pos) {
    // …
}
@fragment void fshader(out vec3 color) {
    // …
}
pipeline my_pipeline {
    @vertex vshader;
    @fragment fshader;
};
```

Testing

210 test files

- Test everything twice for both backends (LLVM and GLSL)
- Can’t call print from shader

```
@gpu int add(int x, int y)
{
  return x + y;
}

@fragment void fshader(out vec3 color)
{
  if (add(17, 25) == 42)
    color = vec3(0., 1., 0.);
  else
    color = vec3(1., 0., 0.);
}
```

Demo time!