Differences between  
OpenGL, OpenGL ES and WebGL

16 July 2015, [Christophe Riccio](mailto:mail@g-truc.net)



[G-Truc Creation](http://www.g-truc.net)

# Table of contents

Table of contents 2

Introduction 4

1. Draw submission 6

1.1. GL\_ARB\_multi\_draw\_indirect 6

1.2. GL\_ARB\_shader\_draw\_parameters 7

1.3. GL\_ARB\_indirect\_parameters 8

1.4. A shader code path per draw in a multi draw 8

1.5. Shader indexed lose states 9

1.6. GL\_NV\_bindless\_multi\_draw\_indirect 10

1.7. GL\_AMD\_interleaved\_elements 10

2. Resources 11

2.1. GL\_ARB\_bindless\_texture 11

2.2. GL\_NV\_shader\_buffer\_load and GL\_NV\_shader\_buffer\_store 11

2.3. GL\_ARB\_sparse\_texture 12

2.4. GL\_AMD\_sparse\_texture 12

2.5. GL\_AMD\_sparse\_texture\_pool 13

2.6. Seamless texture stitching 13

2.7. 3D memory layout for sparse 3D textures 13

2.8. Sparse buffer 14

2.9. GL\_KHR\_texture\_compression\_astc 14

2.10. GL\_INTEL\_map\_texture 14

2.11. GL\_ARB\_seamless\_cubemap\_per\_texture 15

2.12. DMA engines 15

2.13. Unified memory 16

3. Shader operations 17

3.1. GL\_ARB\_shader\_group\_vote 17

3.2. GL\_NV\_shader\_thread\_group 17

3.3. GL\_NV\_shader\_thread\_shuffle 17

3.4. GL\_NV\_shader\_atomic\_float 18

3.5. GL\_AMD\_shader\_atomic\_counter\_ops 18

3.6. GL\_ARB\_compute\_variable\_group\_size 18

3.7. Multi compute dispatch 19

3.8. GL\_NV\_gpu\_shader5 19

3.9. GL\_AMD\_gpu\_shader\_int64 20

3.10. GL\_AMD\_gcn\_shader 20

3.11. GL\_NV\_vertex\_attrib\_integer\_64bit 21

3.12. GL\_AMD\_ shader\_trinary\_minmax 21

4. Framebuffer 22

4.1. GL\_AMD\_sample\_positions 22

4.2. GL\_EXT\_framebuffer\_multisample\_blit\_scaled 22

4.3. GL\_NV\_multisample\_coverage and GL\_NV\_framebuffer\_multisample\_coverage 22

4.4. GL\_AMD\_depth\_clamp\_separate 22

5. Blending 23

5.1. GL\_NV\_texture\_barrier 23

5.2. GL\_EXT\_shader\_framebuffer\_fetch (OpenGL ES) 23

5.3. GL\_ARM\_shader\_framebuffer\_fetch (OpenGL ES) 23

5.4. GL\_ARM\_shader\_framebuffer\_fetch\_depth\_stencil (OpenGL ES) 23

5.5. GL\_EXT\_pixel\_local\_storage (OpenGL ES) 24

5.6. Tile shading 25

5.7. GL\_INTEL\_fragment\_shader\_ordering 26

5.8. GL\_KHR\_blend\_equation\_advanced 26

5.9. GL\_AMD\_blend\_minmax\_factor 27

6. Stencil 28

6.1. GL\_AMD\_shader\_stencil\_export 28

6.2. GL\_AMD\_stencil\_operation\_extended 28

6.3. GL\_AMD\_shader\_stencil\_value\_export 28

7. Rendering pipeline 29

7.1. GL\_INTEL\_conservative\_rasterization 29

7.2. GL\_AMD\_vertex\_shader\_layer 29

7.3. GL\_AMD\_vertex\_shader\_viewport\_index 30

7.4. GL\_AMD\_transform\_feedback3\_lines\_triangles 30

7.5. GL\_AMD\_transform\_feedback4 30

7.6. GL\_AMD\_occlusion\_query\_event 30

7.7. WGL\_AMD\_gpu\_association and WGL\_NV\_gpu\_affinity 31

8. Hardware rings and tasks parallelism 32

Conclusions 33

References 34

# Introduction

The reality of the ecosystem is complex.

# 1. Texture differences

## 1.1. Compressed texture formats

Blablabla.

## 1.2. Unsized [texture](http://www.opengl.org/registry/specs/ARB/multi_draw_indirect.txt) formats

Blablabla.

## 1.3. [Texture](http://www.opengl.org/registry/specs/ARB/multi_draw_indirect.txt) alpha formats

Blablabla.

## 1.4. [Texture](http://www.opengl.org/registry/specs/ARB/multi_draw_indirect.txt) half formats

Blablabla.

## 1.5. [Texture](http://www.opengl.org/registry/specs/ARB/multi_draw_indirect.txt) 16 bit norm formats

Blablabla.

## 1.6. [Texture](http://www.opengl.org/registry/specs/ARB/multi_draw_indirect.txt) float and half filtering

Blablabla.

## 1.7. [ETC](http://www.opengl.org/registry/specs/ARB/multi_draw_indirect.txt) texture formats

## Blablabla.

## 1.8. [Texture](http://www.opengl.org/registry/specs/ARB/multi_draw_indirect.txt) conversion

Blablabla.

## 1.9. BGRA texture format

Blablabla.

## 1.10. [Texture](http://www.opengl.org/registry/specs/ARB/multi_draw_indirect.txt) swizzle

Blablabla.

|  |
| --- |
| layout(binding = INDIRECTION) uniform indirection {  int Transform[MAX\_DRAW];  } Indirection;  layout(binding = TRANSFORM0) uniform transform {  mat4 MVP[MAX\_DRAW];  } Transform;  layout(location = POSITION) in vec3 Position;  layout(location = TEXCOORD) in vec3 Texcoord;  out gl\_PerVertex {  vec4 gl\_Position;  };  out block {  vec2 Texcoord;  } Out;  void main(){  Out.Texcoord = Texcoord.st;  gl\_Position = Transform.MVP[Indirection.Transform[gl\_DrawIDARB]] \* vec4(Position, 1.0);  } |

Listing 2.1.1: Use sample of gl\_DrawIDARB to use a different matrix per draw in a multi draw call

## 1.11. Seamless cubemap [texture](http://www.opengl.org/registry/specs/ARB/multi_draw_indirect.txt)

Blablabla.

## 1.12. Non power of two texture

Blablabla.

# 2. Buffer differences

## 2.1. Buffer target

Blablabla.

## 2.2. Mapped buffer

Blablabla.

# 3. Framebuffer differences

## 3.1. glDrawBuffer

## 3.2. glDrawBuffer(GL\_NONE)

## 3.3. glDrawBuffers(1, &GL\_BACK)

## 3.4. GL\_DEPTH\_STENCIL\_ATTACHMENT

## 3.5. Framebuffer read

## 3.6. Framebuffer sRGB enable

## 3.7. Rendering to float textures

## 3.8. Invalidate framebuffer

# 4. Shader differences

## 4.1. Precision qualifiers

# 5. Misc differences

## 5.1. GL\_MAX\_VERTEX\_UNIFORM\_VECTORS vs GL\_MAX\_VERTEX\_UNIFORM\_COMPONENTS

## 5.2. Multiple transform feedback buffers

## 5.3. Version and extension queries

# 6. Window system differences

# 5. Blending

# Conclusions

# References