****BlendOp  当这个指令存在，则Blend 指令失效****

上一篇文章博主把Shader的Blend指令介绍完了，这一篇我们来看一下BlendOp。BlendOp是指把原像素与Gbuffer中的像素进行某些运算后再渲染出来。这里博主也像上篇那样，把所有的运算指令全部列出来一一说明。（ps:毕竟博主写的是二手资料，建议英语基础好的去读官方文档）（这里博主把原像素称为src,GBuffer中的像素称为dst）

Add src + dst

Sub src - dst

RevSub dst - src

Min 取src和dst较小的一方

Max 取src和dst较大的一样

LogicalClear 全部赋值为0，就是透明

LogicalSet 全部赋值为1，就是黑色

LogicalCopy 复制原像素的值，相当于前面的Blend One Zero

LogicalCopyInverted 把src的值按位取反后渲染

LogicalNoop 渲染dst中的像素，相当于前面的Blend Zero One

LogicalInvert 吧dst中的值按位取反后渲染

LogicalAnd src & dst

LogicalNand ! (src & dst)

LogicalOr src | dst

LogicalNor ! (src | dst)

LogicalXor src ^ dst

LogicalEquiv ! ( src ^ dst)

LogicalAndReverse src & !dst

LogicalAndInverted !src & dst

LogicalOrReverse src | !dst

LogicalOrInverted !src | dst

还是老样子，我们来做个测试:

Shader "Custom/BlendTest"

{

Properties

{

\_MainTex ("Texture", 2D) = "white" {}

}

SubShader

{

BlendOp Sub

Pass

{

CGPROGRAM

#pragma vertex vert

#pragma fragment frag

#include "UnityCG.cginc"

struct appdata

{

float4 vertex : POSITION;

float2 uv : TEXCOORD0;

};

struct v2f

{

float2 uv : TEXCOORD0;

float4 vertex : SV\_POSITION;

};

v2f vert (appdata v)

{

v2f o;

o.vertex = UnityObjectToClipPos(v.vertex);

o.uv = v.uv;

return o;

}

sampler2D \_MainTex;

fixed4 frag (v2f i) : SV\_Target

{

fixed4 col = tex2D(\_MainTex, i.uv);

return col;

}

ENDCG

}

}

}

这里会发现游戏物体没变化，想想也是当然的，GBuffer中现在没有缓存的像素，及全部是0，即使相减后也没有影响，所以就没有变化。这里博主不在做其他其实，有兴趣的读者可以尝试一样别的运算。