Object Glitch Shader

Documentation

Available in Unity Asset Store: http://u3d.as/1Lzz

Discord: https://discord.gg/FGm8QS4

www.inanevin.com

inanevin@gmail.com

1 Introduction

Thank you for purchasing Object Glitch Shader. This documentation contains information about the shader package, compatibility, installation & usage as well as the detailed information about the various shaders included in the package. If you have any questions, comments, feature requests or are in need of help, you can contact me using my email address or you can join the Discord server. Please do rate and review on Asset Store!

1.1 Render Pipelines

Object Glitch Shader is purely based on Shader Graph, and does not support Unity's default renderer. It is supported on Universal and HD render pipelines. Even though officially it has not been released under Lightweight, it can easily be ported into an LWRP project.

1.2 Compatibility

Object Glitch Shader is dependent on render pipeline and shader graph packages, you need to have the corresponding packages with versions above the submitted ones:

- Minimum Unity 2019.3.0b11 and Universal Render Pipeline 7.1.1 & Shader Graph 7.1.1
- Minimum Unity 2019.3.0b11 and High Definition Render Pipeline 7.1.1 & Shader Graph 7.1.1
- Minimum Unity 2019.2.0f1 and Lightweight Render Pipeline and HD Render Pipeline 6.9.0 & Shader Graph 6.9.0

2 Installation

Before installing the package, please make sure that your Unity project contains one of the render pipelines with at least the versions mentioned above, as well as the Shader Graph package along with it.

2.1 Importing Package & Extending Contents

After you have imported the Unity package, please navigate to Assets > InanEvin > ObjectGlitchShader > RP Packages folder. Within this folder, you will find 2 different .unitypackage files, one for URP projects and one for HDRP projects. You can pick one and delete the other. Double-click on the desired package and it will import the rest of the contents needed. Please do not install both package files, as it will cause conflicts and null references. If you accidentally do so, or install the wrong render pipeline contents, please do remove the asset completely (remove the folder InanEvin) and re-import from the Asset Store.

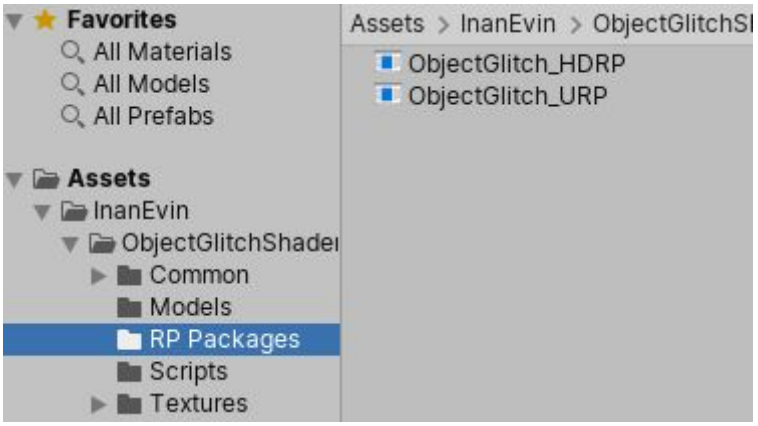

Figure 1 - Folder structure showing where to import the rest of the content

3 Usage

After importing the package and installing the corresponding RP packages, you can navigate to Assets>InanEvin>ObjectGlitchShader>ExampleScenes and open the ObjectGlitchExample scene. There you will find different objects with various materials using the variations of shaders included in the package with a variety of settings. Shader graphs for unlit shaders can be found under Assets>InanEvin>ObjectGlitchShader>Common folder, while shader graphs for lit shaders (URP or HDRP) can be found under Shaders folder. You can create a new material, and navigate to the Shader Graph submenu, where you will find a variety of object glitch shaders.

4 Shader Types

The actual glitch effect on an object's surface is provided by using different masking techniques. Different types of masks can be applied easily to the modularly created Shader Graph structure, and these masks will drive where the glitch effect and vertex deformation is applied. Initially, two types of masks are included with the package; Band and Sphere. These effects have their unlit versions and different lit versions for Universal and HD Render Pipelines.

Along with mask specific parameters, all glitch effects share some common settings that are used to create the actual glitching effect (cellular noise). You can tweak the noise speed, cell division amount and vertex deformation on all shaders.

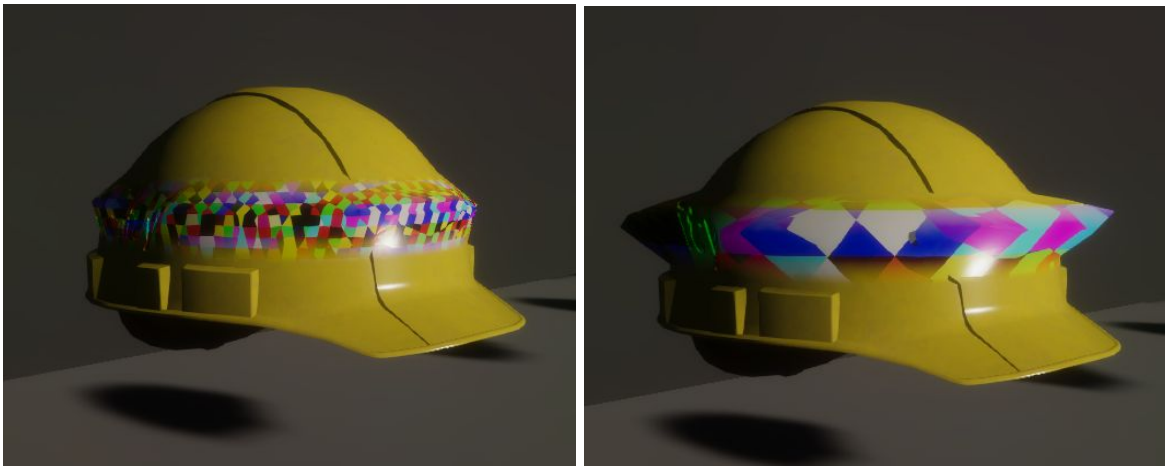

Figure 2 - Band Glitch - 0.5 cell division and 2 vertex deformation(left), 2 cell division and 5 vertex deformation(right)

4.1 Band Glitch

Band glitch provides a band-like mask on the object in order to process the glitch effect and the vertex deformation. The band continuously moves along a designated axis over the object, and it's size, speed and movement length can be tweaked easily within the inspector.

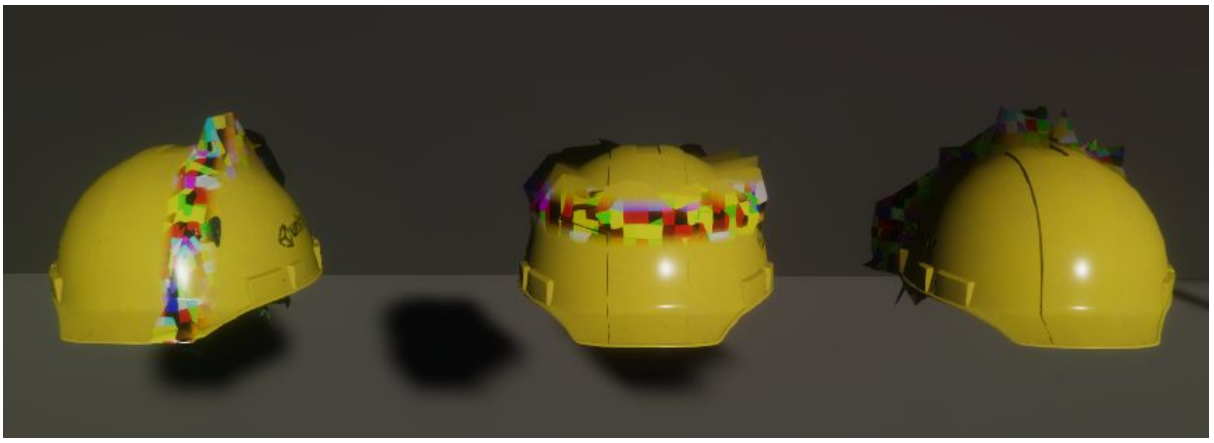

Figure 3 - Band Glitch on 3 different axes (X, Y, Z).

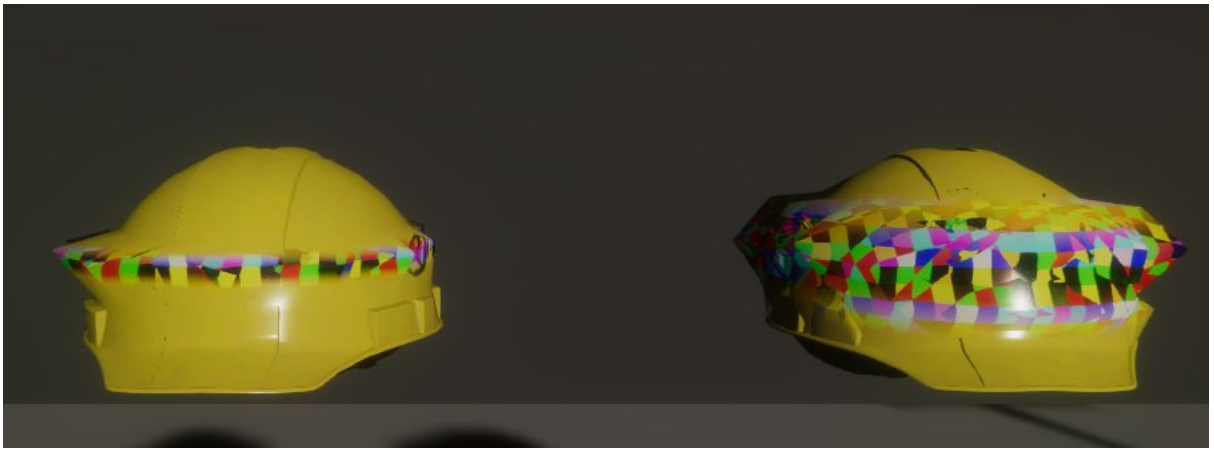

Figure 4 - Band Glitch with different transition lengths & deformation strengths.

4.2 Sphere Glitch

Sphere glitch has the same structure as the band glitch, only difference being it uses a random sphere mask instead of a band mask. It basically works by placing a sphere mask on the object's surface. This area covered by the mask will have the glitch effect and vertex deformation on it, even emission color if desired. The location of the area along the object's surface will randomly be changed depending on a speed parameter. The shape of this area, the randomization speed as well as it's radius are easily tweakable.

Figure 5 - Sphere Glitch with different sphere radius on the same mesh

4.3 Unlit Shaders

The unlit shaders can be found under Assets>InanEvin>ObjectGlitchShader>Common folder. There you will find BandGlitchUnlit and SphereGlitchUnlit shader graph files, both can be used in any of the render pipelines. Unlit shaders provide the corresponding masks (Band or Sphere) with the glitch effect, along with vertex deformation, but does not support emissive glitch area.

4.4 Lit Shaders

Lit shaders provide the glitch effect, vertex deformation and additionally emissive colors on the glitched area. The lit shaders vary between the render pipeline that is being used.

4.4.1 URP Lit Shaders

For Universal Render Pipeline, there are only 2 lit shaders (Band and Sphere), which allow you to input Albedo, Normal and Metallic map on the object.

4.4.2 HDRP Lit Shaders

For HD Render Pipeline, there are 4 lit shaders. 2 of them are for Band Glitch and the other 2 are for the Sphere Glitch. Each effect have two versions, full and simple.

Band/SphereGlitchFull does include every feature that the default HDRP Lit shader includes, from mask mapping to detail mapping. So you can basically use every feature of default HDRP Lit shader with the addition of the glitch effect. If you do not want or need the HDRP Lit features, you can use Band/SphereGlitchSimple, which works like a simple lit shader, in which you can input Albedo, Normal and Metallic maps.

5 Shader Params

All shaders provided have some amount of exposed parameters that are tweakable. Some of these are common amongst all shader types, meanwhile some are shader specific. You can find the glitch related parameters below. All shaders also have some extra parameters that are not related to the glitch effect, like Albedo, Metallic and Normal maps, their strengths and/or colors.

5.1 Common Parameters

The common parameters are the ones that are concerned with how the glitch effect itself is processed, the noise randomization of the glitch and the alpha settings of the affected object.

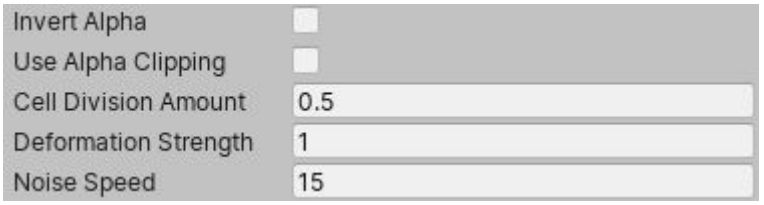

Figure 6 - Common parameters amongst all shaders

- **Invert Alpha:** Determines whether to invert the alpha clip value (subtracting it from 1) when alpha clipping is enabled.
- **Use Alpha Clipping**: Determines whether to use alpha clipping or not. When in use, the glitched area will be solid, while rest will be transparent for sphere glitch (you can invert this), for the band glitch the rest of the object from the banded area will be transparent.

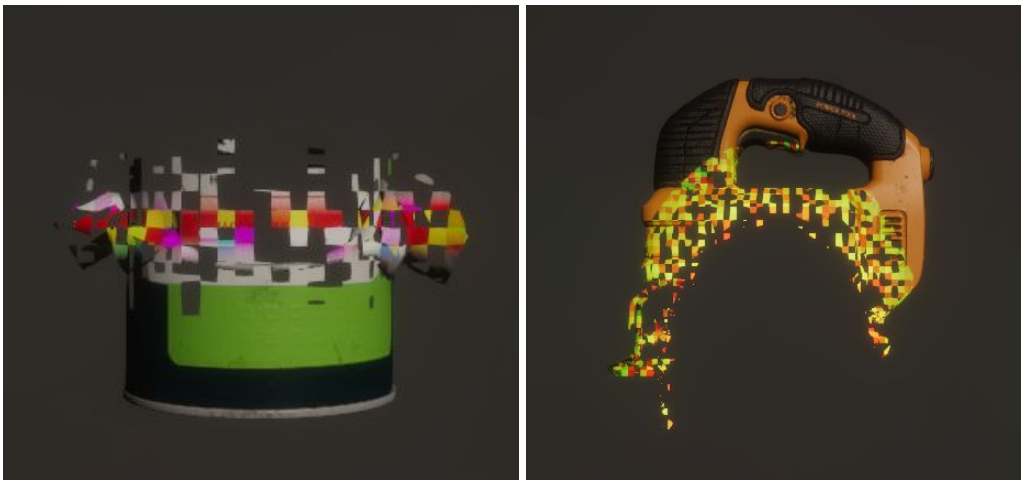

Figure 7 - Alpha clipped band glitch (left), alpha clipped & inverted sphere glitch (right)

- **Cell Division Amount:** Determines how small are the divided square cells within the glitch effect. See Figure 2 for demonstration.
- **Deformation Strength:** Determines how powerful vertex deformation is while applying the glitch effect. See Figure 2 for demonstration.
- **Noise Speed:** The cellular noise in the glitch effect has a constant speed which randomizes the locations of the square cells. This parameter defines that speed.

The parameters below are common only for lit shaders, not the unlit ones.

- **Glitch Emission Color:** The emission color of the glitch effect (the band itself or the sphere area). It is HDR white with 0 intensity by default.

5.2 Band Glitch Parameters

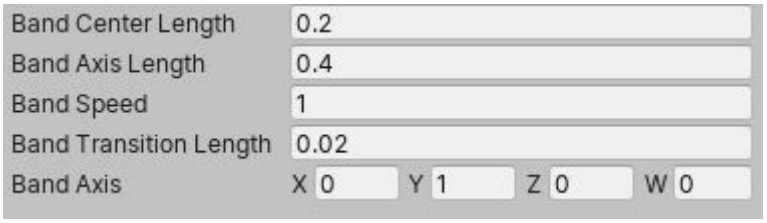

Figure 8 - Band glitch parameters

- **Band Center Length:** The distance that the band starts away from the center of the object. So if the center length is 0, the band will be right on the center of the object. (The center is dependent on the object's pivot, so if the pivot is not the middle of the object, the center will not be either.) Assume that object has a unit length of 0.5 horizontally, so setting the center as 0.25 will make the band start from the right edge of the object. Then you can set the axis length as 0.5 or -0.5 depending on the object's pivot orientation, to make a band that traverses the object completely.
- **Band Axis Length:** The distance that the band travels on single direction (depends on whether this value is + or -) along the target axis from the starting position.
- **Band Speed:** Determines how fast the band traverses the object.
- **Band Transition Length:** Determines the width of the band glitch.
- **Band Axis:** Determines which axis band traverses along. It can be one of three; horizontal (1,0,0), vertical (0,1,0) and depth-wise (0,0,1). The axes have a priority order, so as long as the X value is 1, it will always traverse horizontally regardless of the other components (e.g 1,1,1 -> horizontal). If the X value is not 1 but the Y value is 1, it will choose the Y axis regardless of others and so on. (e.g 0.1, 1, 25 -> vertical)

5.3 Sphere Glitch Parameters

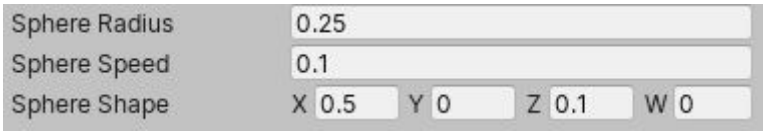

Figure 9 - Sphere glitch parameters

- **Sphere Radius:** Determines the size of the random sphere that causes the glitch effect along the object's surface.
- **Sphere Speed:** Determines how fast the sphere is randomized around the object's surface.
- **Sphere Shape:** Determines the shape of the random sphere that is being placed around the object's surface.

5.4 Parameter Reference Table

You can use the table below to reference and change the values of the parameters by any C# script.

Common Parameters

Parameter Name	**Reference String**	**Parameter Type**
Invert Alpha	_InvertAlpha	Boolean
Use Alpha Clipping	_UseAlphaClipping	boolean
Cell Division Amount	_CellDivisionAmount	float
Deformation Strength	_DeformationStrength	float
Noise Speed	_NoiseSpeed	float
Glitch Emission Color **(Lit Only)**	_GlitchEmissionColor	Color

Band Glitch Parameters

Parameter Name	**Reference String**	**Parameter Type**
Band Center Length	_BandCenterLength	float
Band Axis Length	_BandAxisLength	float
Band Speed	_BandSpeed	float
Band Transition Length	_BandTransitionLength	float
Band Axis	_BandAxis	Vector3

Sphere Glitch Parameters

Parameter Name	**Reference String**	**Parameter Type**
Sphere Radius	_SphereRadius	float
Sphere Speed	_SphereSpeed	float
Sphere Shape	_SphereShape	Vector3

6 Shader Graph Structures In Detail

The structure for the shader graph is very similar in all of the shaders. There are 3 main processes; mask generation, glitch generation and color nodes for Albedo, Normal and Metallic maps. Only for HDRP Full shaders there exists a subgraph HDRPCore, which provides all the functionality that the default HDRP Lit shader has.
Object Glitch Shader is divided into multiple sub-graphs to provide modular structure and extendible functionality.
When you navigate to Assets>InanEvin>ObjectGlitchShader>Common you will find 2 sub folders consisting of subgraphs. Under the NoiseSubgraphs folder you will find Noise1D and Noise3D subgraphs. These provide random noise generation with outputs of float and Vector3, respectively. These noises are used to generate cellular glitch effects.
Inside the GlitchSubgraphs folder, you will find the common subgraphs used to generate the core of the glitch effect.

- **Deformation:** This subgraph provides simple deformation on the input vector, to be used as an input for the vertex position property of master nodes.
- **RandomCell:** Generates random cellular structure and adds noise on top of that to be used by Glitch subgraph.
- **Glitch:** The actual subgraph that creates the glitch effect. It uses Deformation and RandomCell to generate vertex output as well as color output, along with alpha ramp results to be used on the master node.
- **MaskBand:** Subgraph that creates the masking effect for the Band Glitch.
- **MaskSphere:** Subgraph that creates the masking effect for the Sphere Glitch.

Please note that it is possible to create similar subgraphs for new kinds of masking, which can easily be plugged into glitch shaders to create new types of effects.

When you look at the actual .shadergraph files, it can be seen that they are just the combination of these various subgraphs. Initially the mask is generated, either by using MaskBand or MaskSphere graph, which then outputs to the Glitch subgraph. After these two

processes are done, their results are combined with additional nodes that colorizes the object in the means of Albedo, Metallic and Normal mappings (More details for HDRP Full).

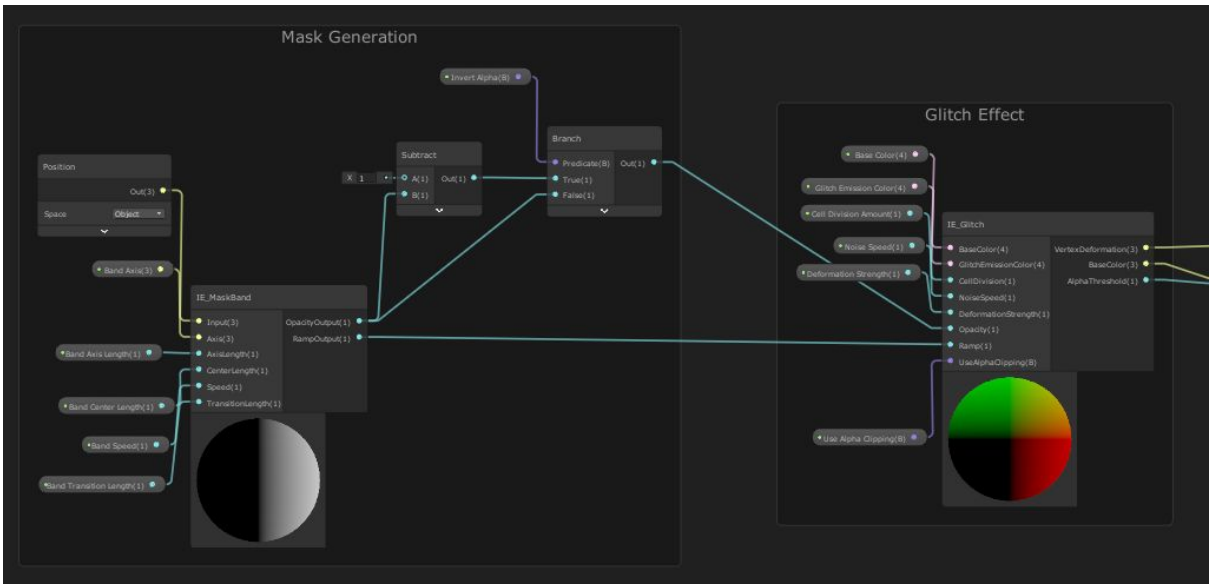

Figure 10 - Generated band mask results connecting to the Glitch Effect sub-graph as inputs.

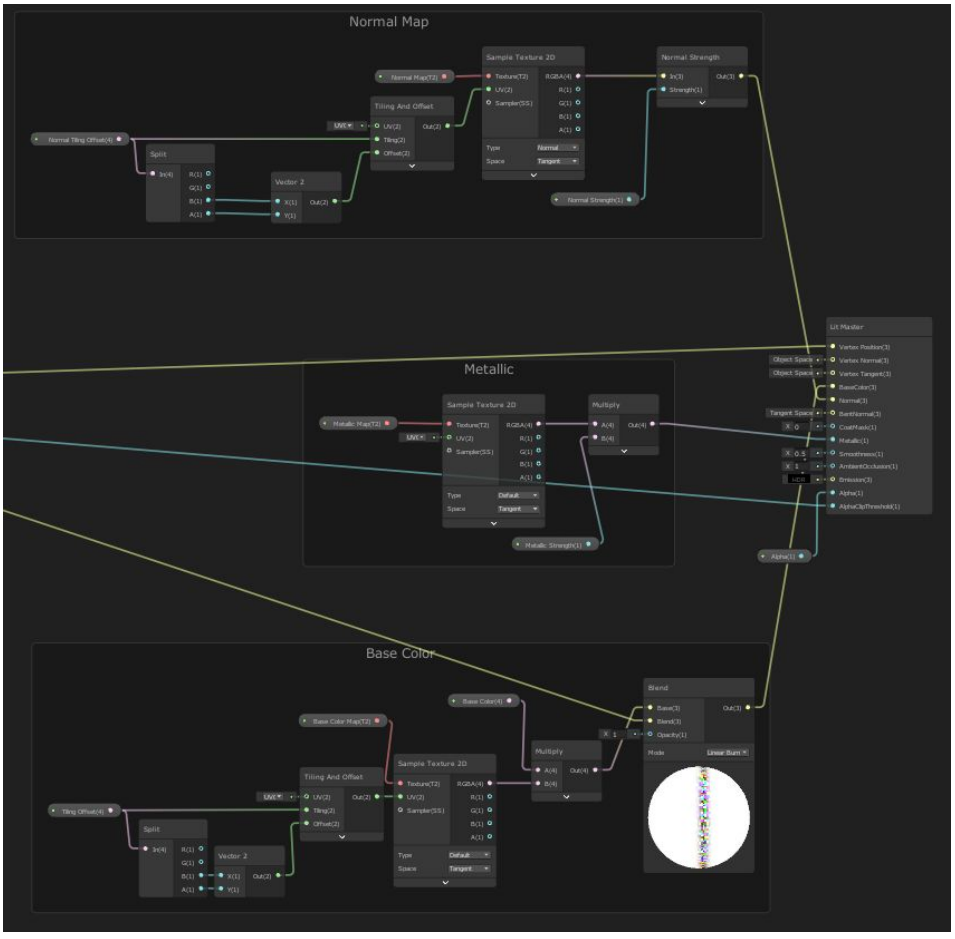

Figure 11 - Results from mask generation and glitch effect connecting to Albedo, Normal and Metallic nodes, which finally connect to the master node to create the desired effect.

7 Important Notes

- Please note that in order for emissive materials to work as intended, you need to have HDR enabled in your project, it's usually under the pipeline settings asset. Additionally, you'd have to have Bloom as a post process working on your scene. On LWRP, you can use Post Processing Stack V2 to achieve Bloom and for URP & HDRP you can use built-in volumes to add the bloom effect.
- Remember to set the normal strength to 0 if you do not have a normal map attached on your materials, failing to do so will mess up the object's lighting.
- HDRP Full materials, the ones with complete HDRP features are only included in HDRP package above Unity version 2019.3.

8 Conclusion

Object Glitch Shader is a package that consists of mainly 2 types of glitch effects, divided into modular subgraph structures for easy extension and tweaking. You can create cellular glitch effects on the object's surface, supported with vertex deformation and emission. The shaders have most of their parameters exposed in order to tweak the effect settings to create a customized result. The package is supported with Universal and HD Render Pipelines, and can easily be ported into Lightweight Render Pipeline. Please do not hesitate to contact me through Discord or by sending me an email to ask any questions regarding the package.