

```
1 using System.Collections.Generic;
2 using UnityEngine;
3
4 /// <summary>
5 /// Logical representation of a BUFFER gate.
6 /// </summary>
7 public class Buffer : Circuit
8 {
9     public Buffer() : this(Vector2.zero) { }
10
11     public Buffer(Vector2 startingPos) : base("BUFFER", 1, 1, startingPos)
12     { }
13
14     /// <summary>
15     /// Returns an output to update if the output has changed due to
16     /// alterations in input power statuses.
17     /// </summary>
18     /// <returns>The list of outputs that should have their connections
19     /// called.</returns>
20     protected override List<Output> UpdateOutputs()
21     {
22         bool outputStatus = Outputs[0].Powered;
23         List<Output> outputs = new List<Output>();
24
25         // BUFFER gate representation
26         Outputs[0].Powered = Inputs[0].Powered;
27
28         if (outputStatus != Outputs[0].Powered || MaterialNotMatching())
29             outputs.Add(Outputs[0]);
30
31         return outputs;
32     }
33
34     /// <summary>
35     /// Checks all outputs to determine if the output node material is not
36     /// matching its power status.<br/><br/>
37     /// This is utilized within custom circuits to force update calls that
38     /// would normally not occur due to the nature of UpdateOutputs().
39     /// </summary>
40     /// <returns>Whether any output material does not match its power
41     /// status.</returns>
42     private bool MaterialNotMatching()
43     {
44         if (Outputs[0].StatusRenderer == null) return false;
45
46         return (Outputs[0].Powered && Outputs
47             [0].StatusRenderer.sharedMaterial !=
48             CircuitVisualizer.Instance.PowerOnMaterial) ||
49             (!Outputs[0].Powered && Outputs
```

...ty Project\Assets\Scripts\Circuits\Starting\Buffer.cs

2

[0].StatusRenderer.sharedMaterial !=

[↗](#)

CircuitVisualizer.Instance.PowerOffMaterial);

41 }

42 }