Node and Nodegraph Connectivity

Introduction

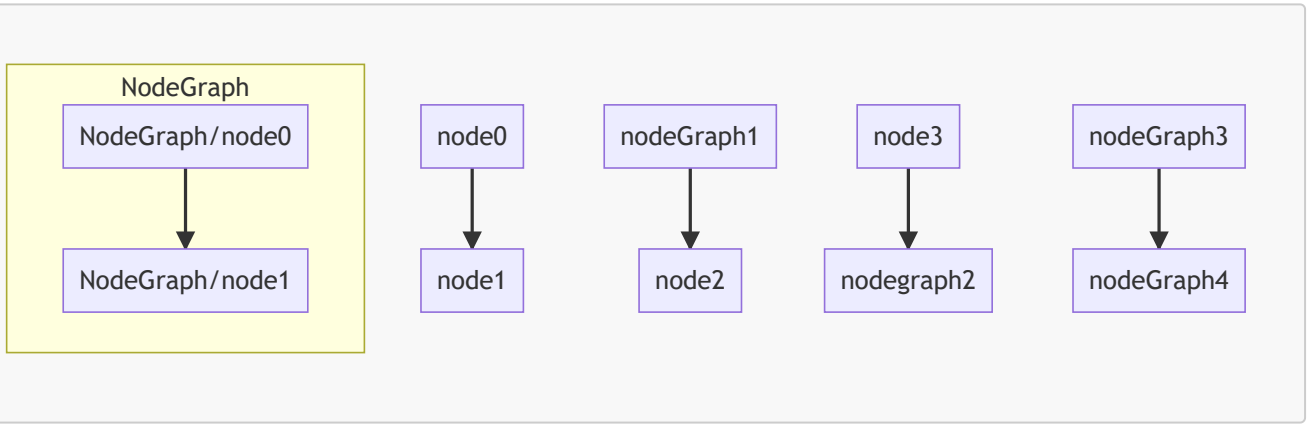
This is a proposal to update the way that connections are specified in MaterialX. To understand the proposal basic background is provided first.

Background: Node Connection Combinations

- A node can be connected to any other node with a nodegraph (node container)
- If you consider a document to also be a node container then nodes can reside a the "top level" of a document as well.
- Any output of a node can be connected to the input of a node or nodegraph if they are of the same type.
- Any output of a nodegraph can be connected the input of a node or nodegraph if the inputs are of the same type.

*Note(): It is valid to have both a value and a connection specified.

Examples

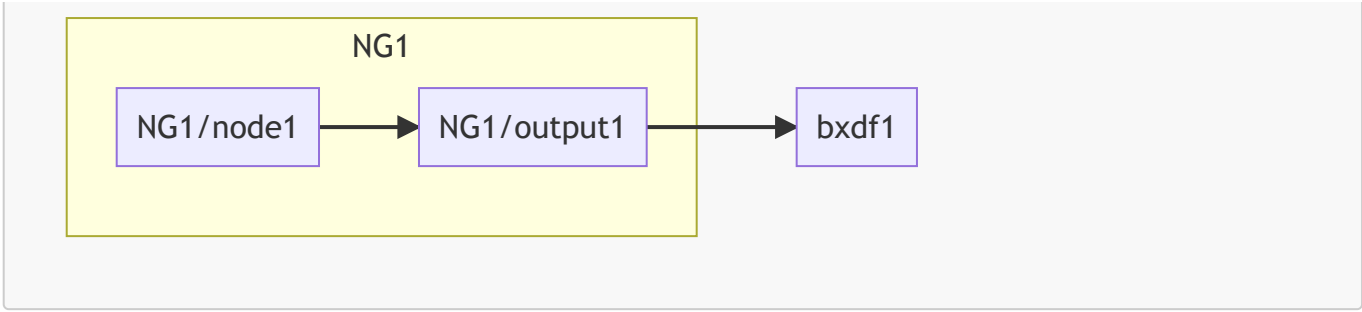


Background: Nodegraph Input / Output Variations

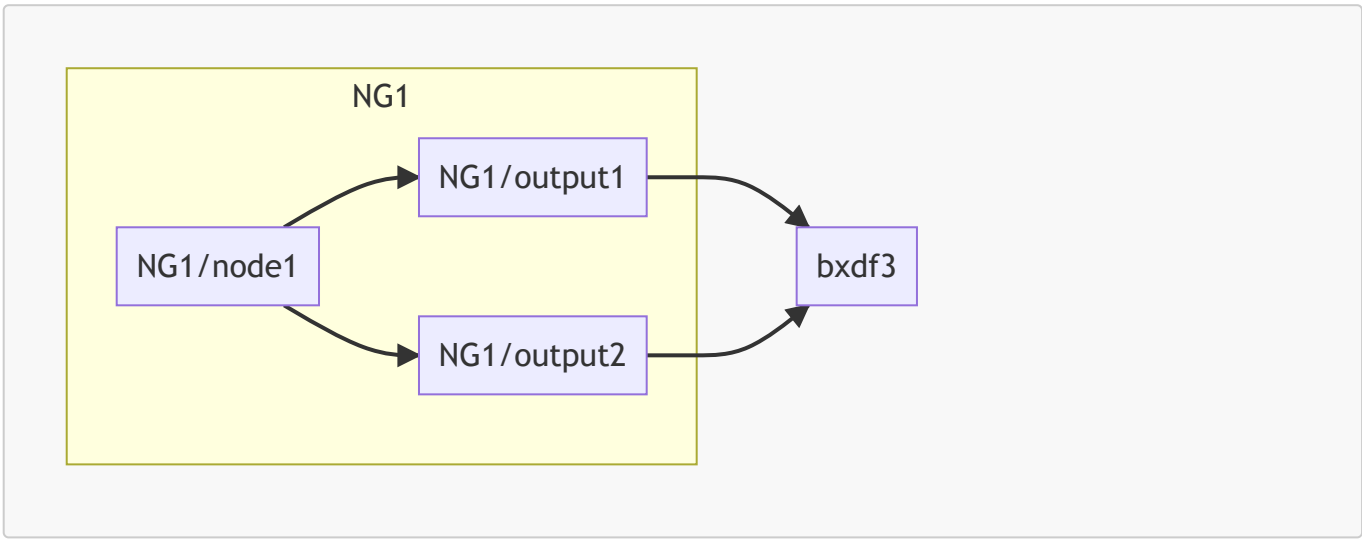
- A `<nodegraph>` can contain a set of nodes.
- Though it is possible have no inputs nor outputs this would be of no practical use.
- Any unconnected input on a nodegraph node can be connected to a nodegraph's input. This is currently via a `interfacename` reference on the node's input.
- An output of a nodegraph node can can be connnected to a nodegraph `<output>`.
- The unique interface "signature" of a nodegraph is defined by it's inputs and outputs.

Examples of Valid Configurations:

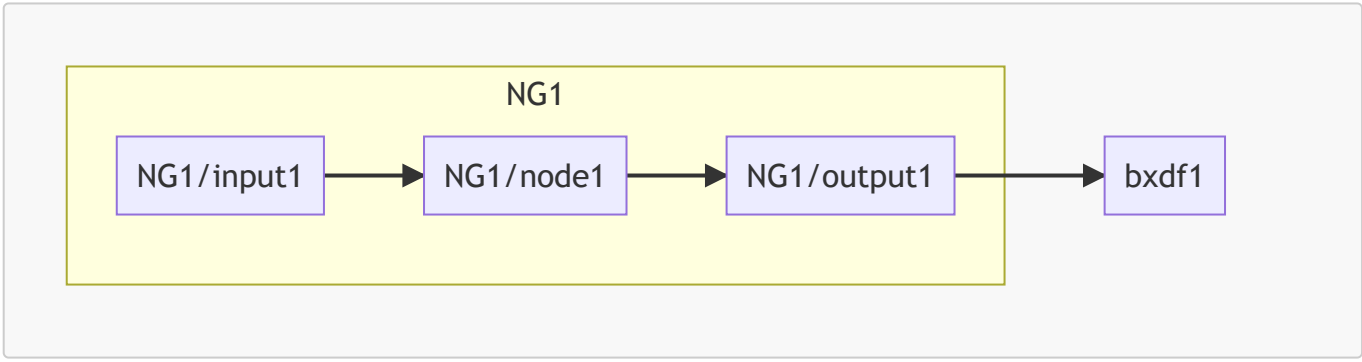
No inputs / Single output



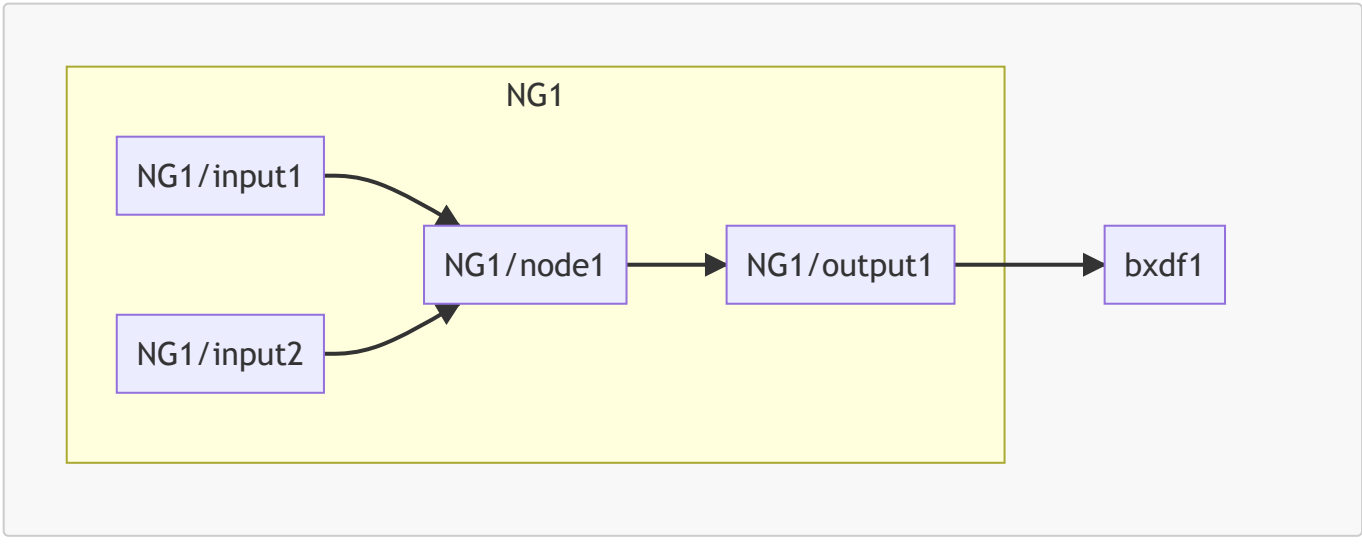
Single input / multiple outputs



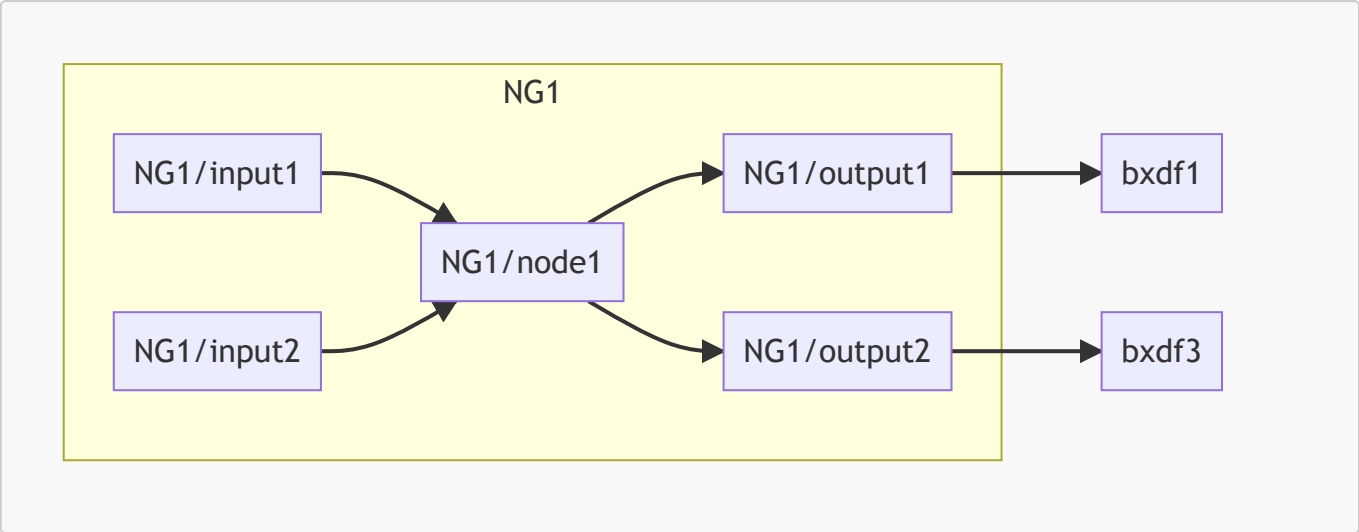
Single input / single output



Multiple inputs / single output

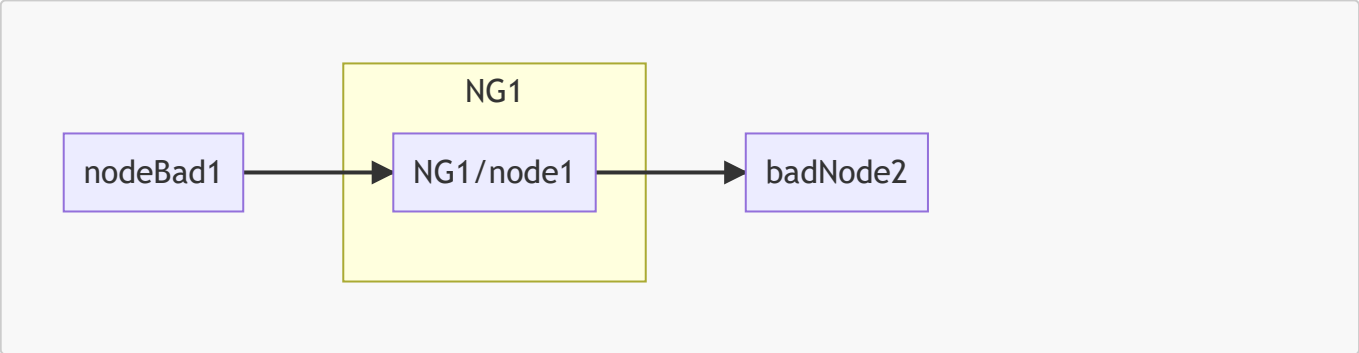


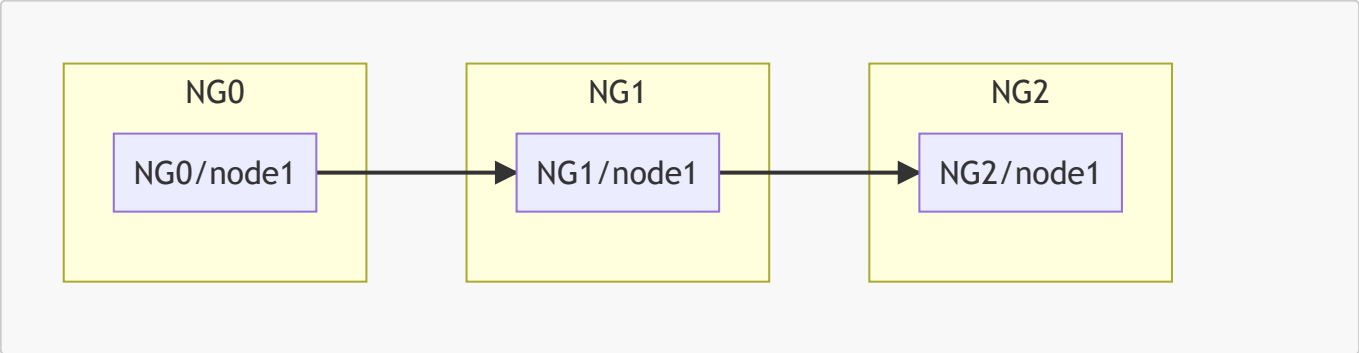
Multiple inputs / multiple outputs



Invalid Configurations

- It is *not* allowed to connect an output of a node or nodegraph directly to an input of a node within a nodegraph. Similarly,
- it is not allowed to connect input of a node inside a nodegraph directory to the output of another node or nodegraph outside the node's nodegraph





Proposed Changes

Connection Syntax

Currently connections are specified on an input using addition attributes:

- `node` if connected to an upstream node output
- `nodegraph` if connected to an upstream nodegraph output
- `interfacename` if connected to the input interface of a nodegraph.

- `output` This must be specified if the upstream node has one or more outputs to avoid ambiguity.
- `channel` Based on the current 1.39 proposal a numeric channel can be specified for an given upstream output.

The proposal is to replace this with a MaterialX path and a single attribute name to representation a connection:

- The attributes:`node`, `nodegraph`, `interfacename` and `channel` will be replaced with a single string attribute named **`connection`**.
  - The assigned value will be a MaterialX path which has been extended to support input interface and channel notation.
- Paths are specified relative to the current scope. This could be within a nodegraph or directly under the root document.
- Thus absolute paths are considered invalid. Syntactically a path starting with a `/` is not valid.
- It is still possible to specify only the path to a upstream node without explicitly specifying it's output if and only if the node only has one output.
  - Otherwise the reference is considered to be ambiguous.
  - A fallback logic of using the first output can still be supported.)
- Inputs which reference an interface input using `interfacename` will use a path notation of this form: `.<input name>`.
  - Note that this means that nodegraph inputs can be connected to node inputs. This makes it orthogonal to how node outputs can be connected to nodegraph outputs. These type of connections are still considered to be invalid if specified outside a nodegraph.
- The 1.39 `channel` attribute (and the existing `channels` attribute are represented as part of the path notation using an array notation: `[#]`. `#` is the channel number 0..N. Where N+1 is the number of channels.
- `<namespace>` can still be supported as part of the path name (as it currently is now). That is it is still allowed to connected between namespaces.

Connections Using A Common Parent

The following are some examples of connections where the connections occur within the parent.

- Connection from node1 in the same parent:

```
<input name="in1" type="color3" connection="node1" />
<!-- or -->
<input name="in1" type="color3" connection="node1.out" />
```

- Connection from channel "1" on node1 in the same parent:

```
<input name="in1" type="float" connection="node1.out[1]" />
```

- Connection from the interface input "albedo" inside a nodegraph:

```
    <input name="in1" type="color3" connection=".albedo" />
```

- Connection from channel "0" of the interface input "albedo" inside a nodegraph:

```
    <input name="in1" type="float" connection=".albedo[0]" />
```

- Connection from node2 to the output interface "result" inside a nodegraph:

```
    <output name="result" type="color4" connection="node2" />
    <!-- or -->
    <output name="result" type="color4" connection="node2.out" />
```

- Connection from channel "3" of node2 to the output interface "resultAlpha" inside a nodegraph:

```
    <output name="resultAlpha" type="float" connection="node2.out[3]" />
```

Connections Between Nodes and Nodegraphs

This example shows:

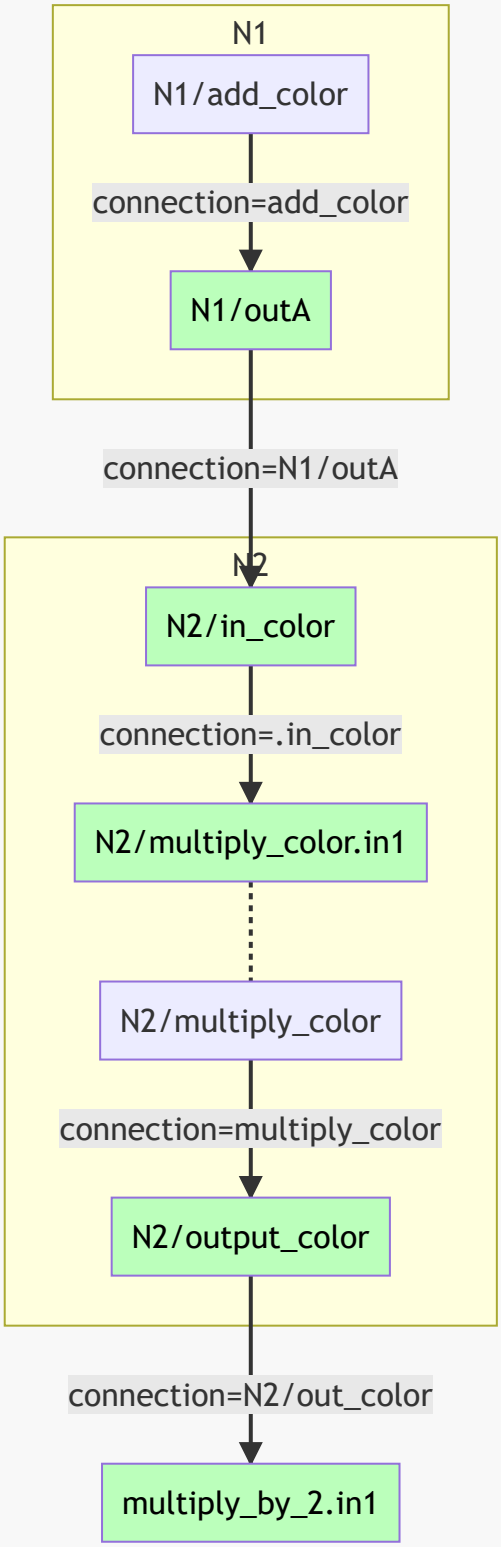
- nodegraph N2's interface input is connected to an upstream nodegraph N1
- nodegraph N2's multiply node input being connected to N2's interface input.
- N1's output being connected to N1's add node's output
- N2's output being connected to N2's multiply nod's output
- node multiply_by_2's input being connected to upstream node N2's output

```
<nodegraph name="N1">
  <add name="add_color" type="color3">
    <input name="in1" type="color3" value="0.1, 0.1, 0.1" />
    <input name="in2" type="color3" value="0.3, 0.4, 0.5" />
  </add>
  <output name="outA" type="color3" connection="add_color" />
</nodegraph>

<nodegraph name="N2">
  <input name="in_color" connection="N1/outA" />
  <multiply name="multiply_color" type="color3">
    <input name="in1" type="color3" connection=".in_color" />
    <input name="in2" type="color3" value="0.5, 0.5, 0.5" />
  </multiply>
  <output name="out_color" type="color3" connection="multiply_color" />
</nodegraph>

<multiply name="multiply_by_2" type="color3">
```

```
    <input name="in1" type="color3" connection="N2/out_color" />
    <input name="in2" type="color3" value="2.0, 2.0, 2.0" />
  </multiply>
```



Restrictions

- Changing scope using something like "../" is not allowed. That is it is not permitted to specify connection to an upstream input or output which is not under the same parent.

Notes

- It is assumed that connection path parsing will occur during document input and output conversion from / to a document.
- This scheme will work with nested nodegraphs assuming that the same connection rules are maintained.