More Mathematica Cool Tips

In[◦]:=
```
Let[Qubit, S]
```

Thread

Suppose you have two lists of inputs and outputs.

In[◦]:=
```
in = Basis[S@{1, 2}]
out = CNOT[S[1], S[2]] ** S[1, 6] ** in
```

Out[◦]=

$$\left\{\left|0_{S_1}0_{S_2}\right\rangle, \left|0_{S_1}1_{S_2}\right\rangle, \left|1_{S_1}0_{S_2}\right\rangle, \left|1_{S_1}1_{S_2}\right\rangle\right\}$$

Out[◦]=

$$\left\{\frac{\left|0_{S_1}0_{S_2}\right\rangle}{\sqrt{2}} + \frac{\left|1_{S_1}1_{S_2}\right\rangle}{\sqrt{2}}, \frac{\left|0_{S_1}1_{S_2}\right\rangle}{\sqrt{2}} + \frac{\left|1_{S_1}0_{S_2}\right\rangle}{\sqrt{2}}, \frac{\left|0_{S_1}0_{S_2}\right\rangle}{\sqrt{2}} - \frac{\left|1_{S_1}1_{S_2}\right\rangle}{\sqrt{2}}, \frac{\left|0_{S_1}1_{S_2}\right\rangle}{\sqrt{2}} - \frac{\left|1_{S_1}0_{S_2}\right\rangle}{\sqrt{2}}\right\}$$

You want to compare the corresponding elements in the two list side by side. Unfortunately, this would not work for obvious reasons.

In[◦]:=
```
in → out
```

Out[◦]=

$$\left\{\left|0_{S_1}0_{S_2}\right\rangle, \left|0_{S_1}1_{S_2}\right\rangle, \left|1_{S_1}0_{S_2}\right\rangle, \left|1_{S_1}1_{S_2}\right\rangle\right\} \to \left\{\frac{\left|0_{S_1}0_{S_2}\right\rangle}{\sqrt{2}} + \frac{\left|1_{S_1}1_{S_2}\right\rangle}{\sqrt{2}}, \frac{\left|0_{S_1}1_{S_2}\right\rangle}{\sqrt{2}} + \frac{\left|1_{S_1}0_{S_2}\right\rangle}{\sqrt{2}}, \frac{\left|0_{S_1}0_{S_2}\right\rangle}{\sqrt{2}} - \frac{\left|1_{S_1}1_{S_2}\right\rangle}{\sqrt{2}}, \frac{\left|0_{S_1}1_{S_2}\right\rangle}{\sqrt{2}} - \frac{\left|1_{S_1}0_{S_2}\right\rangle}{\sqrt{2}}\right\}$$

Thread is useful in such cases.

In[◦]:=
```
Thread[in → out] // TableForm
```

Out[◦]//TableForm=

$$\left|0_{S_1}0_{S_2}\right\rangle \to \frac{\left|0_{S_1}0_{S_2}\right\rangle}{\sqrt{2}} + \frac{\left|1_{S_1}1_{S_2}\right\rangle}{\sqrt{2}}$$

$$\left|0_{S_1}1_{S_2}\right\rangle \to \frac{\left|0_{S_1}1_{S_2}\right\rangle}{\sqrt{2}} + \frac{\left|1_{S_1}0_{S_2}\right\rangle}{\sqrt{2}}$$

$$\left|1_{S_1}0_{S_2}\right\rangle \to \frac{\left|0_{S_1}0_{S_2}\right\rangle}{\sqrt{2}} - \frac{\left|1_{S_1}1_{S_2}\right\rangle}{\sqrt{2}}$$

$$\left|1_{S_1}1_{S_2}\right\rangle \to \frac{\left|0_{S_1}1_{S_2}\right\rangle}{\sqrt{2}} - \frac{\left|1_{S_1}0_{S_2}\right\rangle}{\sqrt{2}}$$

These are typical examples.

In[◦]:=
```
Thread[f[{1, 2, 3, 4}]]
```

Out[◦]=
```
{f[1], f[2], f[3], f[4]}
```

In[◦]:=
```
Thread[f[{1, 2}, {a, b}]]
```

Out[◦]=
```
{f[1, a], f[2, b]}
```

MapThread

Suppose you have two lists.

In[ ]:=
```
Let[Complex, a, b]
aa = Array[a, 5]
bb = Array[b, 5]
```

Out[ ]=

$\{a_1, a_2, a_3, a_4, a_5\}$

Out[ ]=

$\{b_1, b_2, b_3, b_4, b_5\}$

You want to construct a new list such that
$\{F[a_1, b_1], F[a_2, b_2], F[a_3, b_3], F[a_4, b_4], F[a_5, b_5]\}$.

In[ ]:=
```
MapThread[F, {aa, bb}]
```

Out[ ]=

$\{F[a_1, b_1], F[a_2, b_2], F[a_3, b_3], F[a_4, b_4], F[a_5, b_5]\}$

In fact, for the particular example above, **Thread** is already enough.

In[ ]:=
```
F[aa, bb]
```

Out[ ]=

$F[\{a_1, a_2, a_3, a_4, a_5\}, \{b_1, b_2, b_3, b_4, b_5\}]$

In[ ]:=
```
Thread[F[aa, bb]]
```

Out[ ]=

$\{F[a_1, b_1], F[a_2, b_2], F[a_3, b_3], F[a_4, b_4], F[a_5, b_5]\}$

However, some times, **F** itself may have some particular meaning for **List** inputs.

In[ ]:=
```
F[x_List, y_] := x * y
```

In such case, Thread does not have a chance to play its role.

In[ ]:=
```
Thread[F[aa, bb]]
```

Out[ ]=

$\{a_1 b_1, a_2 b_2, a_3 b_3, a_4 b_4, a_5 b_5\}$

However, **MapThread** works correctly.

In[ ]:=
```
MapThread[F, {aa, bb}]
```

Out[ ]=

$\{F[a_1, b_1], F[a_2, b_2], F[a_3, b_3], F[a_4, b_4], F[a_5, b_5]\}$

Through

Suppose that we have a list of qubits.

In[ ]:=
```
SS = S[{1, 2, 3, 4}, $]
```
Out[ ]=

$\{S_1, S_2, S_3, S_4\}$

We want to convert this list to a new list of Pauli X operators on all qubits in the list.

In[ ]:=
```
SX = S[{1, 2, 3, 4}, 1]
```
Out[ ]=

$\{S_1^x, S_2^x, S_3^x, S_4^x\}$

Note that once `S` is declared as a qubit,

In[ ]:=
```
S[1, $]
```
Out[ ]=

S_1

In[ ]:=
```
S[1, $][1]
```
Out[ ]=

S_1^x

In[ ]:=
```
% // InputForm
```
Out[ ]//InputForm=
```
S[1, 1]
```

But, unfortunately, `SS[1]` would not work.

In[ ]:=
```
SS[1]
```
Out[ ]=

$\{S_1, S_2, S_3, S_4\}[1]$

In[ ]:=
```
Through[SS[1]]
```
Out[ ]=

$\{S_1^x, S_2^x, S_3^x, S_4^x\}$

This is a typical example.

In[ ]:=
```
Through[{a, b, c, d}[x]]
```
Out[ ]=
```
{a[x], b[x], c[x], d[x]}
```

Summary

Functions

- `Thread`
- `MapThread`

- `Through`

- Table
- `Apply`
- `Map, MapThread`
- `@, @@, @@@`
- `f@x, f[x], x//f`

Related Links

- S. Wolfram (2017), “An Elementary Introduction to Wolfram Language,” 2nd edition (2017).
- The Wolfram Language: Fast Introduction for Math Students
- The Wolfram Language: Fast Introduction for Programmers