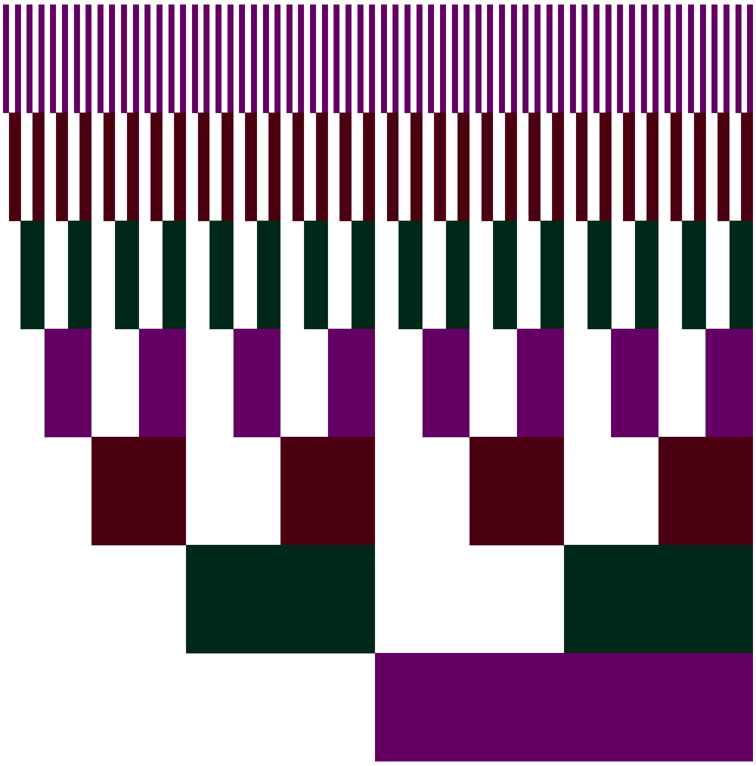
The trick

I really like tricks with binary numbers. This one I was using to test digital systems.



So, were does this come from? Is this generated by some **slow** recursive algorithm? No! This complete binary tree is generated by this code:

```c
if (col & (1 << line)) != 0 {
    *pixel = c[line % c.len()];
}
```

How it works

So, how does this work? Well, let's count binary numbers.

```
111
110
101
100

011
010
001
000
```

The leftmost bit changes every four lines, the middle bit every two lines and the rightmost every other line. From this pattern it's clear that every combination of the three bits will be generated once, for this reason and because it's very easy to generate I was using it to test my digital circuit.

But how do we generate these numbers? Just count from 0 up to 2^n - 1, in this case from 0 to 7:

```
000 = 0
```

```
001 = 1
010 = 2
011 = 3
100 = 4
101 = 5
110 = 6
111 = 7
```

The tree

This is also a complete binary tree. If you think about it as a decision tree, it may be easier to see the reason why. The first bit can be either true of false, for each case the second bit can be true or false too; if one is true, it is highlighted, otherwise it’s black. The number of different leaves is exactly the number of all possible binary strings with `H` bits, where `H` is equal to the height of the tree.