

## Re: BitBox PRNG

**Source:** <http://www.derkeiler.com/Newsgroups/sci.crypt/2003-08/2447.html>

---

**From:** J. Campbell (*mango\_maniac\_at\_yahoo.com*)

**Date:** 08/30/03

Date: 30 Aug 2003 04:44:58 -0700

Michael Amling <nospam@nospam.com> wrote in message news:<2OV3b.17100\$  
> > *the mixing scheme I used is the oft-maligned rule30 1-D automata using*  
> > *cyclic boundary conditions.*  
>  
> *Does a rule30 1-D automaton admit of a brief description of its*  
> *effect on either abcd efgh ijkl mnop or aeim bfjn cgko dhlp?*  
>  
> --Mike Amling

Mike, The rule 30Automata Wolfram's nomenclature for a nearest-neighbor 1D automata. Rule30 is as follows (top line is the neighborhood at tn, bottom line is state of center block at tn+1).

```
111 110 101 100 011 010 001 000
0 0 0 1 1 1 1 0
```

This is called Rule 30 because 00011110 is binary 30. There are 256 possible such nearest neighbor automata, and rule 30 and rule 45 can both be used as PRNGs. Anyway...back to Rule30...it is equivalent to "make the center block on the next line equal to the center block ORed against the right block, and the result of the operation XORed against the left block".

It's important to update the whole automata at once or else it's trivial to go backwards.

Anyway...this automata is one-way, because it loses information about the blocks that are ORed together at each step. It is possible to calculate possible predecessor states, but this calculation becomes daunting as the size of the automata increases, and as the number of not-known lines/number of steps between known states of the system increases.

so...to answer you question considering a 4-element automata, abcd --> a'b'c'd' using wrapping boundary conditions

where:

a' = d XOR (a OR b)

$$b' = a \text{ XOR } (b \text{ OR } c)$$

$$c' = b \text{ XOR } (c \text{ OR } d)$$

$$d' = c \text{ XOR } (d \text{ OR } a)$$

and similarly,

aeim  $\rightarrow$  a'e'i'm' using wrapping boundary conditions

where

$$a' = m \text{ XOR } (a \text{ OR } e)$$

$$e' = a \text{ XOR } (e \text{ OR } i)$$

$$i' = e \text{ XOR } (i \text{ OR } m)$$

$$m' = i \text{ XOR } (m \text{ OR } a)$$

Joe