FEEDBACK SHIFT REGISTER RANDOM NUMBER GENERATOR

Example of Feedback Shift Register (FSR)

4 bits: 1 2 3 4 5

state: | 1 | 1 | 0 | 1 | ...

Produce the 5th bit with the following rule:

$b_5 = b_{5-p} \hat{} b_{5-q}$, where $\hat{}$ means exclusive OR operation. Then shift the register one bit to the left.

The maximum number of values that FSR can have is $2^4 = 16$ in this example. With a good choice of p and q the register can produce a sequence of period $2^4 - 1 = 15$ before it repeats itself (the value of all bits 0 is excluded, because no other state can be produced from it).

Let us try $(p, q) = (4, 3)$, $b_5 = b_1 \hat{} b_2$.

1 2 3 4 5

| 1 | 1 | 0 | 1 | 0 |

$1 \hat{} 1 = 0$ →

← by 1 bit

| 1 | 0 | 1 | 0 |

0 1 0 1

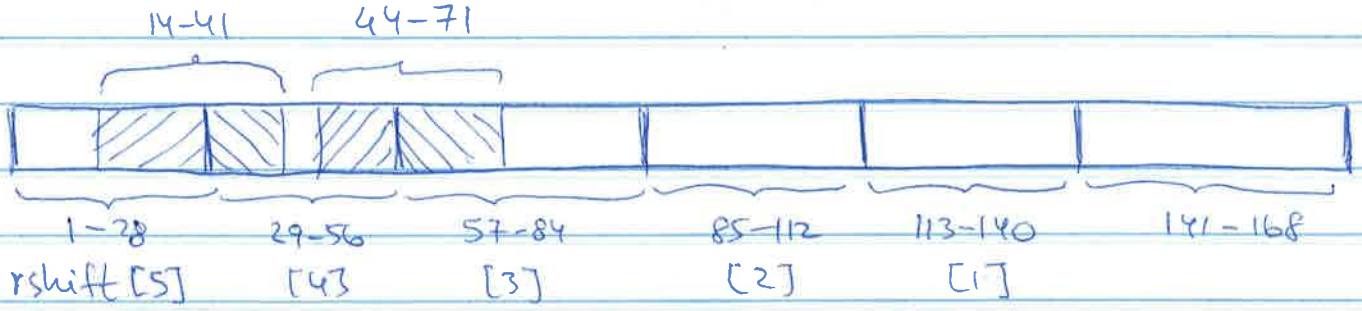
1 0 1 1

...

This choice of (p,q) produces a sequence with period 15.
Other choices may lead to shorter periods (e.g. $(3,2)$ gives period 7), so (p,q) must be chosen carefully.

We can now extend this scheme to work with groups of bits, for instance, produce random numbers with the length of 28 bits. The period for 28-bit FSR is $2^{28}-1$ (with a good choice of (p,q)).
To increase the period we can work with a longer FSR, but take only 28 bits for the random numbers.

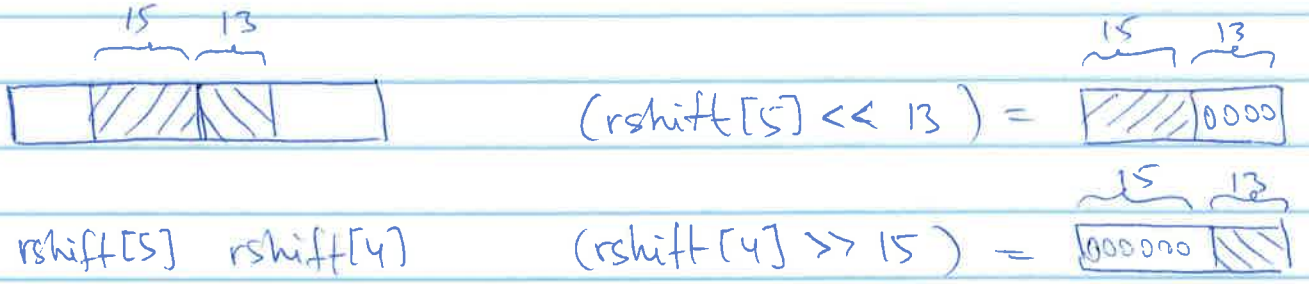
In the example that we will consider we take 140-bit FSR to produce 28-bit long random numbers, so the period is $2^{140}-1$.



The rshift[1] to [5] array stores the state of the FSR. Our choice of the pair is $(p,q) = (97, 127)$.

$$b_n = b_{n-97} \hat{} b_{n-127}, \quad n = 141, \ldots, 168$$

For the XOR operation we need bits starting with $141 - 97 = 44$ and $141 - 127 = 14$



<< denotes bitwise shift to the left, and >> to the right in the C language.

Once we get needed bits in place, we can combine them with | (OR) operation:

t_2 = (rshift[5] << 13) | (rshift[4] >> 15)

Similarly, t_1 = (rshift[4] << 15) | (rshift[3] >> 13)

We then combine t_1 and t_2 with XOR operation and mask 28 bits (remember, we use float, i.e. 32 bits to store the FSR state).

Then we shift the register to the left and produce the floating point number in [0, 1] by dividing by 2^{28}.