Compact, digital pseudo-random number generator

Davide Anguita, Stefano Rovetta, and Rodolfo Zunino

A general purpose, easy to implement random number generator is presented. It features an eight-bit word size, good statistical properties, and repeatability of the generated sequence. This circuit has been included in the design of many projects, ranging from neural networks to cryptographic systems.

Introduction: Many applications require the realization of a random number generator featuring some desirable properties [1]; often such a device is at the very heart of a system. Examples of applications in the electronic field include cryptography, for which the period length serves as a minimum protection versus brute force attacks, and initialization of parameters for training of neural networks. We require that the system generates a sequence of numbers featuring good statistical properties, i.e. as uniform as possible. In the case of cryptography reproducibility is also crucial; in general, when the system requires a repeatable sequence, a pseudo random sequence is needed, that is, a sequence generated by a single number (the *seed*) by a chaotic process simulating randomness, as opposed to real noise generators. We would like to obtain the longest possible period before the repetition of the sequence. Moreover, when the generator algorithm is planned to be translated into hardware, the preference will be for the simplest circuit.

The proposed scheme features a good statistical behavior, easy scalability to increase the length of the period, very simple realization with digital hardware, and the ability to produce also the zero value as part of the output sequence, thus implementing a random variable with values in [0, 255) as often assumed by applications, without requiring additional computations that would reduce the compactness of the circuit.

Pseudo-Random Number Generators: The proposed generator is based on the model of *linear recursive generation* (LRG) [2], particularly well-suited for hardware implementation. The basic building block is a shift register. The method of LRG is based on a linear recursive sequence of bits, given by the following definition:

$$X_k = C_1 X_{k-1} + C_2 X_{k-2} + \dots + C_p X_{k-p}$$

where X_k is the k-th bit of the generated sequence, and the p coefficients C_i are binary constants which determine the behavior of the generator. The linear recursive sequence is initialized with a set of p values $\{ X_1, \dots , X_p \}$. The first $p - 1$ constants C_i take on values in $\{ 0, 1 \}$ and $C_p = 1$.

Since the next bit X_k depends only on the past p bits, the maximum period n we can hope to achieve will be $n = 2^p - 1$. There holds the following [1]:

Theorem. The period n has its maximum possible value $2^p - 1$ if and only if the polynomial

$$f(x) = 1 + C_1 x + C_2 x^2 + \dots + C_p x^p$$

is prime on the field of the polynomials with coefficients in {1,0}.

A desirable property of the sequence generated by such a set of coefficients is that it is statistically balanced (it averages to 1/2: SX_k/n = 1/2).The circuit implementation of such a simple generator is straightforward, and it is shown in Fig. 1. It involves only a shift register and an exclusive-or computing the sum of the p bits $C_1 X_{k-1} \dots C_p X_{k-p}$.

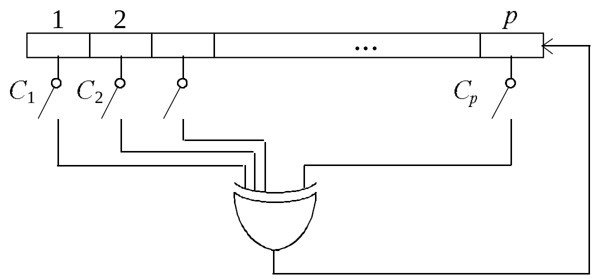

Fig. 1 *Linear recursive generator.*

This basic scheme, producing a single bit stream with pseudo-random behavior, can be used to build up a generator for a stream of multi-bit words. If a word length of b bits is required, the first b bits of the shift register can be used (if $b < p$), or another shift register of appropriate length can be appended to the output of the single-bit circuit to accumulate enough bits to form a complete word. In either case, the period of the sequence is reduced to $(2^p - 1)/b$ words. A better generator can be designed using b parallel LRG. A *shift matrix* C_{ij} is obtained by considering the b rows of p shift coefficients each. It can be shown that the period of this circuit is still 2^p-1 if the shift matrix has full rank. The drawback is a more complex realization.

Usually the full period of 255 is not sufficient for most of the realizations, even for those requiring 8-bit words. Hence the basic LRG should be improved by making its period longer. It is not very difficult to obtain such a goal, but we must be cautious when doing this because of the important requisite of statistical uniformity of the sequence; in the random processes terminology, the sequence should have an impulsive autocorrelation. In designing the generator we should also take into account the actual word size b required. In fact, some schemes are better behaving, if analyzed from a statistical standpoint, when realized with a larger number of bits. In general it is harder to design a generator with good independence properties for a smaller word size. For those reasons, once a generator has been designed, it is a good practice to simulate it in software and to analyze its output with a statistical independence test, like the standard Chi-Squared test.

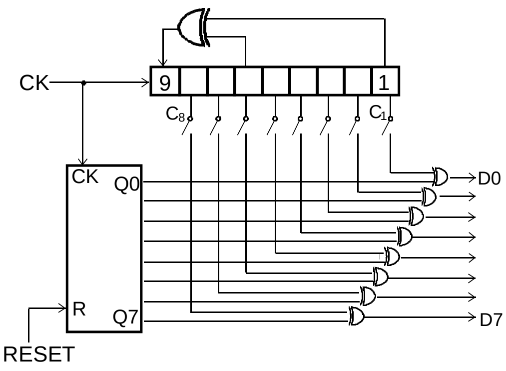

Fig. 2 *The proposed circuit.*

Implementation: The circuit is shown in Fig. 2. It is based on the idea of combining two generator blocks in such a way that their periods interact to form the longest possible combination. This means that if the two blocks have periods n_1 and n_2 respectively, we wish the combination to yield a period of $n=n_1 n_2$. This property can be achieved when the two periods are mutually prime.

The first block is a LRG with 9 bits, initialized with an 8-bit seed. The last bit is loaded with a fixed value (selected at design time). The period is thus $2^9 – 1 = 511$.

If the constant initialization bit is set to 1, the generated sequence includes also the value 0, that is consequently a legitimate value for the seed. This property may be required, for instance, in those applications in which the seed is user-selectable, and thus it cannot be guaranteed that a particular value will never be introduced.

The second block does not need to be another pseudo-random generator, but only an element introducing a de-correlation among the successive bits of the sequence; for this purpose it is sufficient to provide an 8-bit counter, with period 256. Since 511 and 256 are mutually prime, the global period of the generator amounts to $256 \cdot 511 = 130816$.

The initialization is made through two 8-bit words, one to set up the counter and the other to provide a seed for the LRG.

The standard c^2 test yielded good results about the uniformity of the sequence. The c^2 parameter remained within the acceptable limits for most of the sequences, generated with different seeds.

Concluding remarks: The proposed circuit has been used in the design of many projects currently under development. Some examples include the VLSI implementation of a Vector Quantization neural network (Plastic Neural Gas [3]), requiring a set of initial random values for training, and a hardware cryptographic system implementing a scaleable fractal algorithm. A more thorough validation of the statistical properties of the sequence is needed, especially for the cryptographic application; however, the selection of a theoretically sound and practically useful testing method is still an open problem.

The proposed circuit is especially well-suited for applications requiring a compact circuit for generating random sequences, as in digital realizations of distributed computing applications [4].

Acknowledgment: The authors thank Marco Cappelli, whose ideas originated this work.

References

1 L'ECUYER, P.: 'Uniform random number generation,' *Annals of Operations Research*, 1994, **53**, pp. 77-120

2 KNUTH, D.E.: 'The Art of Computer Programming: Seminumerical Algorithms,' Vol. 2, 2nd edition (Addison Wesley, 1981)

3 RIDELLA, S., ROVETTA, S., and ZUNINO, R.: 'Plastic Neural Gas for Adaptive Vector Quantization,' submitted to *IEEE Transactions on Neural Networks*

4 PARODI, G.C., RIDELLA, S., ZUNINO, R., 'Using chaos to generate keys for associative noise-like coding memories', *Neural Networks*, 1993, **6**, No. 4, pp. 559-572

 8 June 1995

Davide Anguita, Stefano Rovetta, and Rodolfo Zunino *(DIBE, University of Genoa, Via all'Opera Pia 11a 16145 Genova – Italy)*