A Fast Method of Generating Digital Random Numbers

By C. M. RADER,* L. R. RABINER and R. W. SCHAFER

(Manuscript received June 12, 1970)

In this article we propose a fast, efficient technique for generating a pseudorandom stream of uniformly-distributed numbers. The arithmetic operations required are an L bit exclusive-or, a rotation, and a shift to update the state of the number generator. With moderately large values of L we have been able to generate sequences of numbers whose periods are quite long (on the order of 2×10^7 *long). Its simplicity of construction, as well as its ability to generate long streams of independent pseudorandom uniformly-distributed integers make this noise generator a worthy candidate for use in high-speed digital systems.*

I. INTRODUCTION

Almost all methods of generating digital random numbers use as input a set of previously generated random numbers which were produced by an iterative arithmetic process (modulo a large integer)—e.g.

$$X_n \equiv F(X_{n-1}, X_{n-2}, \cdots, X_{n-J}) \bmod N \qquad n = 0, 1, \cdots$$

with initial conditions

$$\begin{aligned} X_{-1} &= C_1 , \\ X_{-2} &= C_2 , \\ &\cdot \\ &\cdot \\ &\cdot \\ X_{-J} &= C_J . \end{aligned}$$

For each new random number, X_n, this arithmetic process is repeated. The integer N and the initial values $C_1, C_2, \cdots, C_J$ are chosen to

* Mr. Rader is with Lincoln Laboratories, Massachusetts Institute of Technology, Lexington, Massachusetts. Lincoln Laboratories is operated with support from the U. S. Air Force.

guarantee a large period of repetition. Methods of the type described above involve a considerable propagation delay, representing at the least one addition or one multiplication time, between the time the nth random number is put into its storage register, and the time at which the $(n + 1)$st random number is available. This delay is not generally a problem in most applications, because computational delays in other parts of most digital systems are far greater than those encountered in generating random numbers. However, it is conceivable that in the future a need will arise for which random numbers must be computed far more rapidly than is now necessary. With such a time in mind we propose the following algorithm, for which the time necessary to produce a new random number is equal to the sum of a flip-flop settling time, and the propagation delay of an exclusive-or gate.

II. THEORY

The algorithm for generating the L-bit random number X_n from the two previous L-bit numbers X_{n-1} and X_{n-2} may be stated as

$$X_n = T_P(X_{n-1} \oplus X_{n-2})$$

where $T_P(\cdot)$ denotes a cyclic rotation of P places to the right and $\oplus$ denotes exclusive-or. The algorithm requires L flip-flops to store X_{n-1} and L flip-flops to store X_{n-2}. Each bit of the new random number X_n is derived by an exclusive-or operation on a pair of corresponding bits in the previous random numbers. These bits are not stored in the bit positions from which they were produced, but instead each new bit is rotated cyclically to the right by P bit positions, with overflow on the right being fed into the left. The process is illustrated in Fig. 1 for $P = 1$. At each cycle, the new random number generated (X_n) is clocked into the lower flip-flops (X_{n-1}); at the same time the number stored in the lower flip-flops (X_{n-1}) is clocked into the upper flip-flops (X_{n-2}). For maximum period, P should be chosen mutually prime to L. Otherwise, the bit rotation may be shown to be composed of several interleaved rotations of shorter words. As seen from Fig. 1, the bit rotation does not constitute a separate hardware operation. In physical terms, the exclusive-or of two flip-flops in corresponding bit positions is clocked into a third flip-flop while one of the pair of flip-flops is clocked into the other. An example of the process is given in Table I for $L = 3$, $P = 2$. (The starting values of $X_{-1} = 000$, $X_{-2} = 001$ are used here.) The period of this generator is 15. It is easily shown that

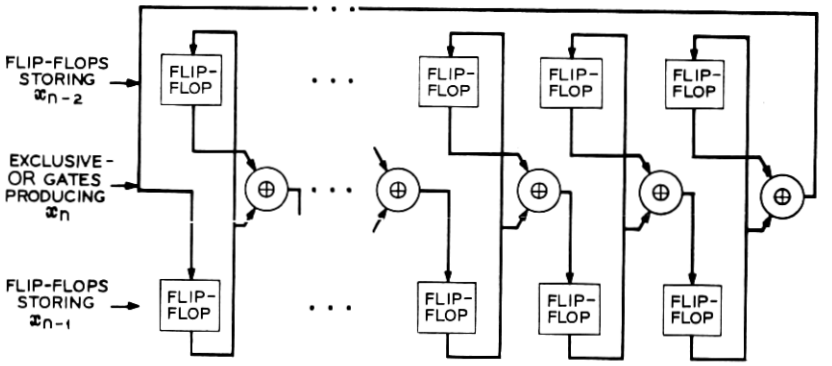

Fig. 1—Schematic diagram showing how the random numbers are generated and stored.

with the output depending on the state of six ($2L$) flip-flops, the maximum theoretical period of any random number generator is 64, or in general (2^{2L}), and the maximum period of any random number generator using only flip-flops and exclusive-or elements is 63, or in general ($2^{2L} - 1$), since the state when all the flip-flops are zero is succeeded only by itself.

Even though the period of the generator of Table I, and the periods

TABLE I—TYPICAL OUTPUT SEQUENCE FOR NOISE GENERATOR WITH $L = 3$, $P = 2$

Clock Number	X_{n-1} Most Recent	X_{n-2} Next Most Recent	X_n New Random Number
0	000	001	010
1	010	000	100
2	100	010	101
3	101	100	010
4	010	101	111
5	111	010	011
6	011	111	001
7	001	011	100
8	100	001	011
9	011	100	111
10	111	011	001
11	001	111	101
12	101	001	001
13	001	101	001
14	001	001	000
(15)	(repeats 000)	(repeats 001)	(repeats 010)

for other values of L, are small fractions of the theoretical maximum, it is possible to choose L to obtain a very long period. It is also possible, as we shall see, to combine the results of several generators to get still longer periods. We have theoretically predicted all the word lengths ($L \leqq 25$) for which very short periods result*, and we have measured the periods associated with the remaining values of L. The longest periods result for word lengths of $L = 11, 13, 17, 19, 22, 23$ and 25. For 25 bits, the longest period results, and is 17,825,775. However, since 2^{25} is about 3.3×10^7, not all the possible 25 bit numbers appear at the output of the generator. The computation of the period assumes that the word is rotated a number of bits mutually prime to the word length (e.g., $P = 1$) and that the starting states are reasonable (e.g., $X_{-1} = 0$, $X_{-2} = 1$). There exist unreasonable starting states, such as all zeros in one word and all ones in the other word, which have much shorter periods than those prescribed, but these can be avoided in all cases by restricting the starting values to 0 and 1.

Table II lists the periods of the generators for values of L from 1 to 25, as well as the factorization of these periods. The factorization is useful in predicting the periods associated with generators made up of two or more of these simple generators with interleaved bits. For example, it is possible to generate a 48 bit random number by interleaving the bits of a 25 bit word with the bits of a 23 bit word. The periods of the two generators may be found to have only the prime factor 3 in common, as seen from Table II. Thus the joint period (the least common multiple) is $\frac{1}{3}$ of the product of the periods of the individual generators, resulting in a period of about 2×10^{13}. Similarly a 24 bit word could be made up of an 11 bit word and a 13 bit word, with a joint period of about 2×10^9. The disparity of prime factors among several interesting cases seems surprisingly fortuitous.

III. TESTS FOR RANDOMNESS

It is clear that a long period does not by itself indicate a good random number generator (e.g. the iteration $r_n = r_{n-1} + 1$ would have a

* It is possible to equate any bit (as a function of time) to the mod 2 sum of the same bit delayed by various amounts. For example, with $L = 3$ the equation for any bit of the 3 bit word is:

$$b_n = b_{n-3} \oplus b_{n-4} \oplus b_{n-5} \oplus b_{n-6}.$$

This equation describes a particular 6 bit shift register with feedback. The analysis of such shift registers is described in Ref. 1. Specifically it is possible to obtain the maximum period associated with a given shift register, and therefore with a given random number generator, by obtaining the factors of certain characteristic polynomials over the Galois Field mod 2.

TABLE II—THE PERIOD AND ITS DECOMPOSITION INTO ITS PRIME FACTORS FOR NOISE GENERATORS WITH VALUES OF L FROM 1 TO 25

L	Period	Factors
1	3	3
2	6	(2) (3)
3	15	(3) (5)
4	12	$(2)^2$ (3)
5	255	(3) (5) (17)
6	30	(2) (3) (5)
7	63	$(3)^2$ (7)
8	24	$(2)^3$ (3)
9	315	$(3)^2$ (5) (7)
10	510	(2) (3) (5) (17)
11	33825	(3) $(5)^2$ (11) (41)
12	60	$(2)^2$ (3) (5)
13	159783	(3) (13) (17) (241)
14	126	(2) $(3)^2$ (7)
15	255	(3) (5) (17)
16	48	$(2)^4$ (3)
17	65535	(3) (5) (17) (257)
18	630	(2) $(3)^2$ (5) (17)
19	14942265	(3) (5) (13) (19) (37) (109)
20	1020	$(2)^2$ (3) (5) (17)
21	4095	$(3)^2$ (5) (7) (13)
22	67650	(2) (3) $(5)^2$ (11) (41)
23	4194303	(3) (23) (89) (683)
24	120	$(2)^3$ (3) (5)
25	17825775	(3) $(5)^2$ (11) (17) (31) (41)

very long period, but would be unacceptable to most users). A better indication of the acceptability of a random number generator is the autocorrelation function of the output of the generator, $R_x(n)$. It is difficult to obtain $R_x(n)$ theoretically for the generators described here; so instead we have estimated $R_x(n)$ by standard techniques for several cases of interest. In Fig. 2 we show the estimated autocorrelation function for the generator with $L = 13$. Approximately $N = 15000$ samples were used in the estimate, and Fig. 2 shows the results for up to 512 delays. It seems clear that there are no irregularities present in the autocorrelation function. The peak values of the autocorrelation function of Fig. 2, for $n \neq 0$, are ± 0.026. It is easily shown that estimates of the autocorrelation function (for $n \neq 0$) tend to be normally distributed random variables with zero mean, and variance of $1/N$. For the data of Fig. 2, the standard deviation, σ, was calculated to be 0.008. Cramer[2] shows that the expected value of the upper extreme of 512 values from a normal population with zero mean, and standard deviation of σ, is approximately 3.25 σ, or 0.026 in this example. The 50 percent

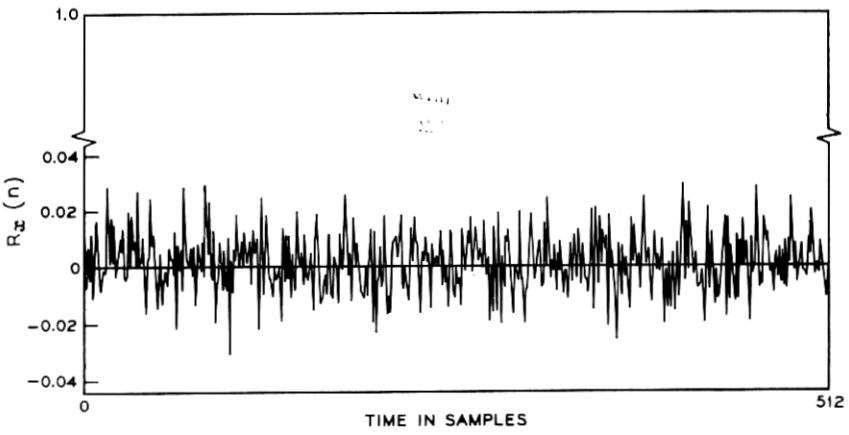

Fig. 2—Autocorrelation function of noise generator with $L = 13$.

confidence interval for the upper extreme is about 0.4 σ wide, thus peak values of the autocorrelation estimates from 0.024 to 0.028 would be quite common. Therefore the peak values of ± 0.026 in Fig. 2 are not inconsistent with the above theoretical results based on a true normal population.

To empirically test the uniformity of the output of the generators for $L = 11$ and $L = 13$, we measured the number of occurrences of each of the 2^L output states during a single period. For both cases all states were present at the output of the generator. In Fig. 3, we show plots for $L = 11$ and 13 of the number of occurrences of each of the 128 cells specified by the 7 most significant bits of the output as a function of cell number. The upper plot shows the measured result for $L = 11$, and the lower plot shows the measured result for $L = 13$. The solid lines across the plots indicate the expected number of occurrences of each state based on uniformity assumptions. The plots of Fig. 3 tend to validate the assertion that the amplitude distribution of the output of the noise generator is uniform for certain values of L.

We have also measured the mean value for the entire sequence for $L = 11$ and $L = 13$ and it is near to 2^{L-1} if the L bits are interpreted as a positive integer. The nearest means obtained were 1024.3169 for $L = 11$ and 4095.8326 for $L = 13$. (Starting values for the two sequences were $X_{-1} = 341$, $X_{-2} = 0$ for $L = 11$, and $X_{-1} = 151$, and $X_{-2} = 0$ for $L = 13$.) Also, it was found that a scatter diagram from the generator with $L = 17$ showed no tendencies to order, such as are common

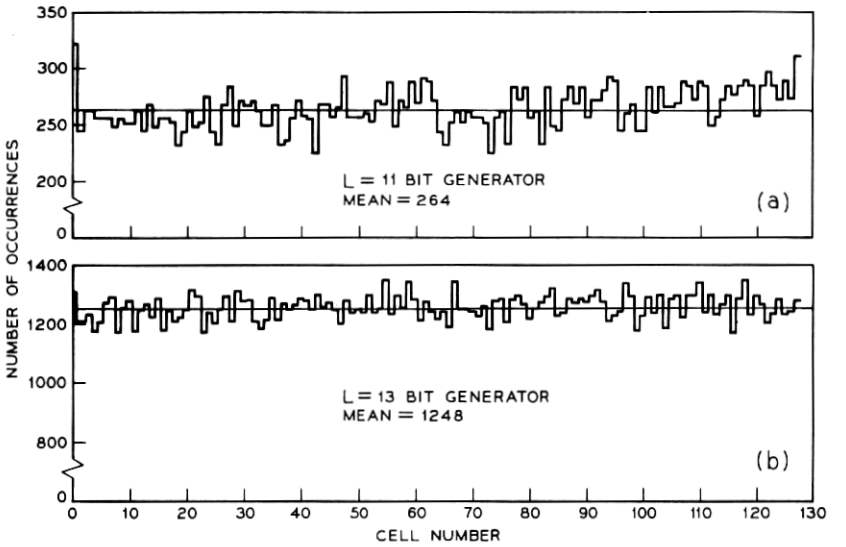

Fig. 3—Measured distribution functions for noise generators with $L = 11$ and $L = 13$.

in the simple multiplicative congruence generators. It is expected that this result would be true for any of the generators with reasonably long periods.

It was stated earlier that it is possible to generate longer random numbers than 25 bits by interleaving bits of shorter generators. Besides the prerequisite that the periods of the individual generators have few or no prime factors in common, it is important that the outputs of the individual noise generators be uncorrelated. To check whether

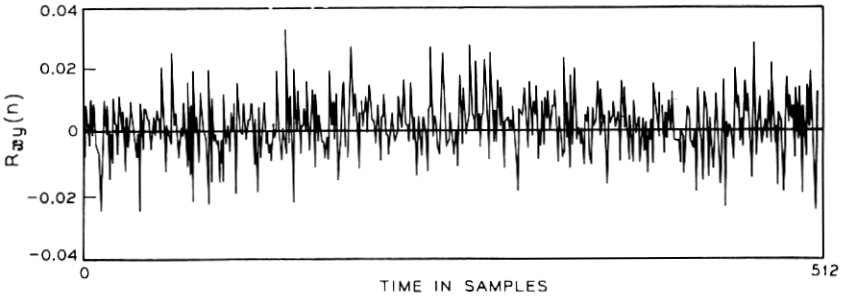

Fig. 4—Cross-correlation function of noise generators with $L = 11$ and $L = 13$.

or not the individual outputs of the noise generators, for the desirable values of L, were uncorrelated, we again used standard statistical techniques to measure the cross correlation function. Figure 4 shows the cross-correlation function, $R_{xy}(n)$, for the case where one input was the output of the generator with $L = 11$, and the other input was the output of the generator with $L = 13$. The cross-correlation function is plotted for delays up to 512 samples, and is again based on approximately $N = 15{,}000$ samples. There are no apparent irregularities seen in this figure, and the peak values of the cross-correlation function, ± 0.027, are quite close to similar peaks observed for the autocorrelation function of Fig. 2, and again consistent with the assumption that these 512 estimates are from a normal population with $\sigma = 0.008$.

The reader should be cautioned that a poor choice of the rotation P can cause an ordering in the pseudorandom outputs which may be harmful for some applications. For example, consider the set of all triples (x_n, x_{n+1}, x_{n+2}) when $P = 1$. If the sign bits (using two's complement integer notation) of x_n and x_{n+1} are the same, the most significant bit of x_{n+2} will always be zero. Thus, if (x_n, x_{n+1}) lies in quadrants 1 or 3, x_{n+2} is constrained to either

$$0 \leqq x_{n+2} < 2^{L-2}$$

or

$$-2^{L-1} \leqq x_{n+2} < -2^{L-2}.$$

A similar constraint results when (x_n, x_{n+1}) lies in quadrants 2 or 4. This effect is eliminated by choosing $P \approx L/2$, but mutually prime to L. We have not considered what ordering might exist in quadruples, quintuples, etc., for various choices of P.

IV. CONCLUSION

In conclusion, we have presented a fast and efficient technique for generating digital random numbers. The simple statistical tests to which we have subjected several of these noise generators indicate they are more than adequate for use in simulation programs for communications systems.

REFERENCES

1. Golomb, S., *Shift Register Sequences,* San Francisco: Holden-Day Inc., 1967.
2. Cramér, H., *Mathematical Methods of Statistics,* Princeton, N. J.: Princeton Univ. Press, 1946, pp. 363–378.