Design of an Analog/Digital Truly Random Number Generator

S. Espejo†, J. D. Martín††, A. Rodríguez-Vázquez†† and J. L. Huertas††
†ATT Bell Labs, 600 Mountain Av., Murray Hill, USA
††Dto. de Diseño Analógico-CNM, Edificio CICA, C/ Tarfia s/n, 41012-Sevilla, SPAIN

Abstract

An analog digital system is presented for the generation of truly random (aperiodic) digital sequences. This model is based on a very simple piecewise-linear discrete map which is suitable for implementation using monolithic analog sampled/data techniques. Simulation results are given illustrating the optimum choice of the model parameters. Circuit implementations are reported for the discrete map using both switched-capacitor (SC) and switched-current (SI) techniques. The layout of a (SI) prototype in a 3μm n-well double poly double metal technology is included.

Introduction

A random number generator is a basic building block in communication and instrumentation application[1]. Also, there are many simulation problems arising both in science and engineering where random numbers are required [2]. These potential applications are wide and interesting enough as to challenge design engineers to conduct research towards the electronic implementation of this kind of signal generators.

The use of maximum cycle feedback-shift-registers (FSR) provided a way for the electronic generation of a signal exhibiting certain pseudorandom properties. This kind of hardware implementation is commonly used for the purpose of testing VLSI circuits, where the unavoidable periodicity of the sequence may not be an important drawback. Typical operating systems of mainframe computers include, on the other hand, software for the generation of random sequences exhibiting "better" characteristics than the ones provided by FSR. However periodicity can be always detected over a more or less long time. Besides, increasing the quality of the sequence requires a tradeoff with the increase of CPU time.

All these drawbacks result as a consequence of the use of purely digital techniques. Periodicity can be avoided by resorting to the use of analog techniques. In particular a very simple random number generator model is proposed in this paper which exploits the chaotic behavior of nonlinear discrete-time systems. This model can be implemented in monolithic form using state of the art sampled-data analogue circuit techniques together with digital circuitry, thus making possible on-chip generation of truly-random signals in combined analog/digital chips.

An architecture for random number generator

Fig.1 shows the conceptual block diagram for a monolithic A/D random number generator. Three different blocks can be observed:

1) The discrete map.
2) The comparator.
3) The digital processor.

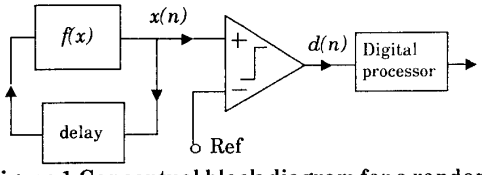

Figure 1:Conceptual block diagram for a random number generator based on a discrete map.

The *discrete map* is the crucial building block for a monolithic random number generator. It has to be designed to provide a *chaotic* (aperiodic) analog signal *x(n)*. Starting from this analog signal, a one-bit digital sequence *d(n)* can be obtained by *comparison* with a reference level. Although for *x(n)* being chaotic the resulting digital sequence is fully aperiodic, some correlation can be usually observed among successive samples of *d(n)*. Besides, different probabilities measures are typically obtained for *one* and *zero* events of *d(n)*. The *digital processor* in Fig. 1 is intended to equalize those probabilities and to eliminate correlations among successive samples.

The discrete map

Starting from an arbitrary initial point *x(0)* in *I*, a discrete-time chaotic signal *x(n)* can be generated by applying the following iterative formula:

$$x(n)=f[x(n-1)] \quad n=1,2,3... \tag{1}$$

which is called a discrete map of the interval *I*.

The implementation of a discrete map is conceptualized in Fig.1. A delay block and a non-linear operator are required. Many non-linear operators are qualified to produce chaos [3,4]. Only some of them are however suitable for monolithic implementation. In particular we will focus here on *piecewise-linear* (PL) functions because they can be very easily and accurately implemented in MOS VLSI [5,6].

Let us start by considering the following discontinuous PL map:

$$f_0(x)=Bx-A\,sgn(x) \tag{2a}$$

where *sgn(x)* holds for the *sign function*:

$$sgn(x)=\begin{cases}1 & x\geq 0\\ -1 & x<0\end{cases} \tag{2b}$$

Parameters *A* and *B* determine the properties of the signal generated by a discrete map based in (2a). Fig.2 illustrates the parameter dependence of this function. Analysis shows that parameter *A* is only a scale factor. On the other hand, different qualitative behaviors can be obtained by changing parameter *B*, namely:

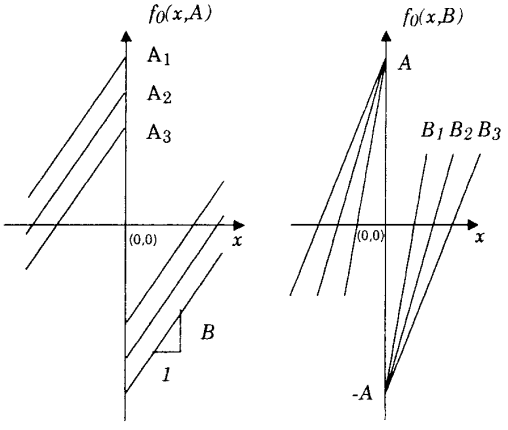

Figure 2: Parameter dependence of the map in (2a).

- For B<1, a stable periodic orbit results, the sequence $x(n)$ oscillating between the values:

$$x_1^* = A/(1+B) \qquad (3)$$
$$x_2^* = -A/(1+B)$$

This is obviously an undesirable situation for the application being considered.

- For $B>1$ There are no stable periodic orbits of any period.

- For $B>2$ there are points inside the region $(-A,A)$ whose orbits diverge so leading to unstable global behavior.This situation is illustrated in Fig.3.

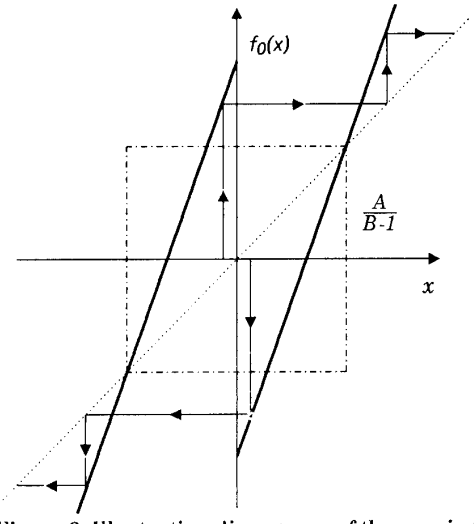

Figure 3: Illustrating divergence of the map in (2a).

Let us concentrate on the range of parameter values where neither stable orbits nor divergent points are found, $(1<B<2)$. Three subintervals can be distinguished for this case inside the interval $(-A,A)$:

$(0,A)$	All the orbits starting in this interval remain confined to it.
$(A,A/(B-1))$	All the orbits starting here map into the subinterval (-A,A)
$(A/(B-1),\infty)$	All the orbits starting here are divergent.

These different properties of the subintervals are illustrated in Fig.4a.

In practical implementation of (2a), design criteria should be provided to avoid the correspondent circuit to be locked at parasitic stable points, caused by the saturation of the active devices. Owing to previous considerations we conclude that this can be achieved by setting the level of the bias sources at a value comprised in the second subinterval above. It is in fact equivalent to transform the original map into the one in Fig.4b. Observe that any orbit eventually going into one of the saturation regions is forced to return to the region of interest $(-A/(B-1),A/(B-1))$.

Observe from Fig.4a that the value of B has to be selected small enough to provide wide lateral safety subintervals: $(A/(1-B),-A)$, $(A,A/(B-1))$. On the other hand, the larger the value of B $(B\leq 2)$ is the more suitable is the signal $x(n)$ for the generation of random numbers. As a matter of fact an ideal Bernouilli shift results for $B=2$ [4].

We hence see that a tradeoff in the values of B is required in any practical design. This tradeoff can be avoided by resorting to a slightly more complex map which is shown in Fig.5 and whose equation is:

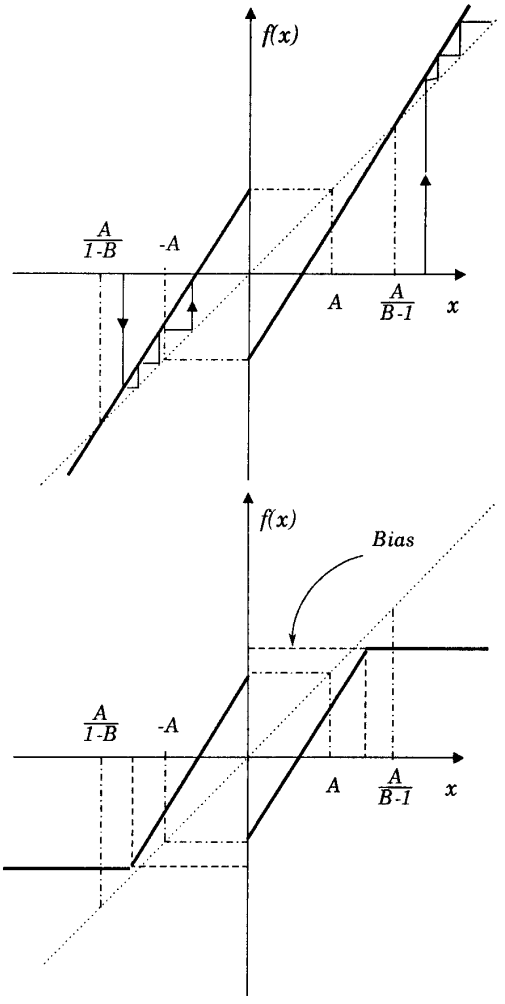

Figure 4: a)Illustrating orbits from the different subintervals; b)Avoiding parasitic stable points by proper bias setting

$$f(x) = \begin{cases} f_0(x) & |x|<A \\ 2x - C\,sgn(x) & |x|\geq A \end{cases} \qquad (4)$$

where $f_0(x)$ is given by (2a) with $B=2$ and C is some constant greater than A. The value of C is proven not to be critical by numerical tests. $C=2A$ seems to be a good value. As it can be seen in Fig.5, points that lie in the region $(A, 2A)$ are *re-mapped* into the region $(0, A)$. Even if the slope B is greater than 2 the map is still stable and apparently exhibits the same behavior as for B less than 2.

Analog sampled-data implementations

The discrete map of (4) can be very easily and accurately implemented in MOS VLSI . Fig.6 shows a two phase stray-insensitive SC circuit. Observe the output of the comparator directly codifies the sign of the chaotic analog sequence $x(n)$. That means there is no need to use the extra comparator in the block diagram of Fig.1.

In order to ensure full compatibility with digital technologies, *switched-current* techniques can be used[8]. A circuit implementation using this technique is shown in Fig.7. Observe only MOS transistors are required. As in the SC circuit, the comparator required for the map directly provides the random digital sequence.

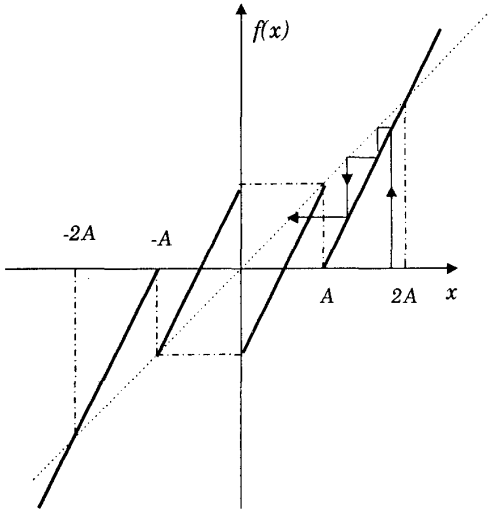

Figure 5: Illustrating the map in (4)

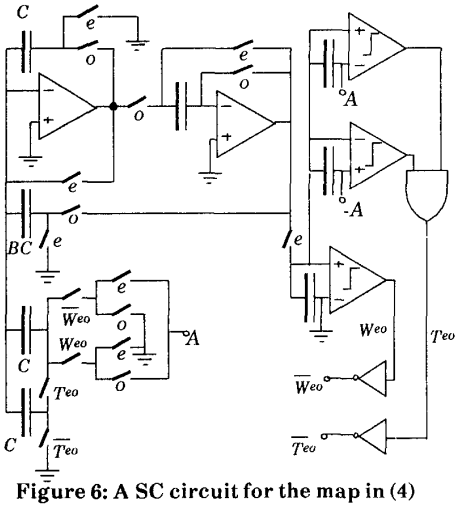

Figure 6: A SC circuit for the map in (4)

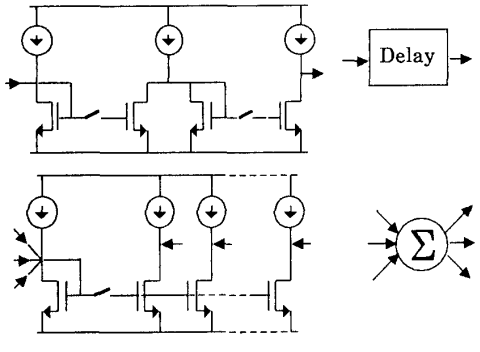

Figure 7a: Basic blocks for a SI map

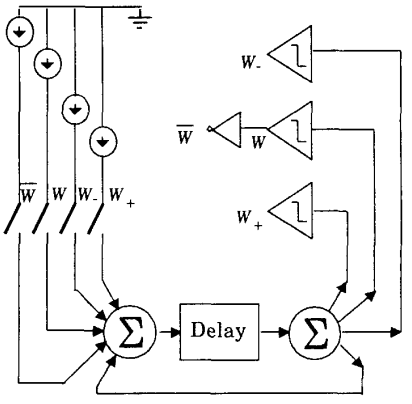

Figure 7b: SI implementation of the map in (4)

Simulation Results and Digital Processor

A lot of simulations have been performed to confirm suitability of the proposed model. In order to analyze *regularity* properties, a random process consisting of a set of around 1000 maps was considered. Parameters in the different maps of the set were randomly modified (taking into account typical technological deviations) around their nominal value. Up to 10^5 iterations were considered to properly analyze stationarity properties. Also, in order to more realistic emulate the behavior of an actual circuit implementation, a gaussian noise was added to the calculated sample of the signal in each iteration step. The simulations were carried out on a 15Mflops CONVEX 220 vectorial computer.

The bits in the sequences were grouped into digital words (both 4 and 8 bits) and the probability of each word was measured. For the map of (2a) with $B=1.85$ (limit value) the deviation from the ideal distribution for words of 4 bits was 46% in the typical case, and 59% in the worst case. On the other hand, for the map (4) with $B=2$, the typical deviation was 12% being 35% in the worst case.

For both maps, the observed deviations can be interpreted as being a consequence of the autocorrelation of the bits in the original sequence. This interpretation suggests a technique to equalize probabilities consisting in decreasing the sampling rate for eliminating the self correlation.The price to be paid for is a decreasing in the operation speed of the circuit which may be not convenient depending on the application.

An alternative method is based on the use of a very simple digital element: the T-bistable. Let us remind the input-output map of such a digital block:

Input \ State	0	1
0	0	1
1	1	0

(5)

Consider now an arbitrary digital sequence driving a T-bistable. Taking (5) into account it is easy to conclude that the digital sequence at the output will exhibit the same properties for *"1"* and *"0"* events. Also if the probabilities of *"1"* and *"0"* were already the same, then all the sequences of 2 bits will have the same probability at the output. To see this keep in mind that the states *"0"* and *"1"* are equally probable because they depend on the parity of the number of *"1"*'s received by the bistable; as long as the input sequence is aperiodic the parity is completely random. Thus, we have that connecting two T-bistables to the digital generator all the sequences of two bits at the output have the same probability.

In general when n T-bistables are connected to the digital generator, all the sequences of n bits have the same probability.

To prove this consider a system consisting of n T-bistables. Given an input sequence $x_i(n)$, there is one and only one possible state of the system for each possible output sequence $x_o(n)$. So, if all the states of the system are equally probable, all the output sequences are equally probable too. Now, all the states of the system are equally probable because the state of the first bistable depends on the parity of the number of *"1"s* received, the state of the second bistable depends on the parity generated by the first (not on the state of it), the third on the second and so on. Again as long as the input sequence is aperiodic, those parities are random and so is the state of the system.

Unfortunately n T-bistables do not guarantee equal probability for words over n bits, but we can use a hardware inplementation algorithms of the kind proposed by Knuth [7] . If we want to generate words of n bits not self-correlated, after the process described above, we fill a RAM of 2^n words, then numbers are picked up from RAM addresses given by words of the sequence delayed a large enough number of cycles. The address used is then refilled with the incoming word and the process continues. A block scheme of the digital processor is shown in Fig.8.

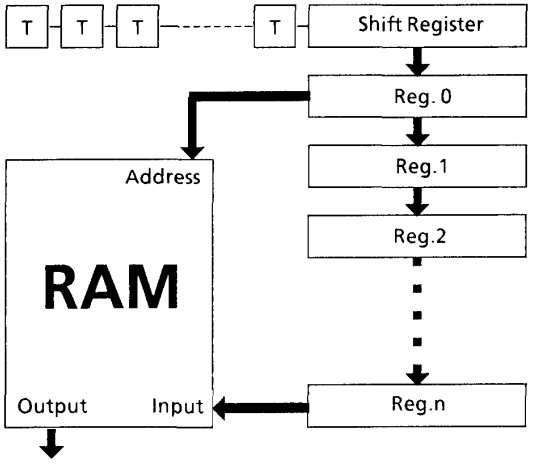

Figure 8: Block scheme of the digital processor.

Discussion of results

Piecewise-linear maps are easy to implement using sampled-data analogue techniques, either SC or SI,and provide a natural way for the generation of truly random number generetors in monolithic form. Several prototypes of Fig.6 and Fig.7 have been designed both in 3μm and 2μm CMOS. In particular Fig.9 shows the layout of a SI prototype for a n-well double poly 3μm CMOS technology. Simulation results for these prototypes are in accordance to the expected theoretical performance.

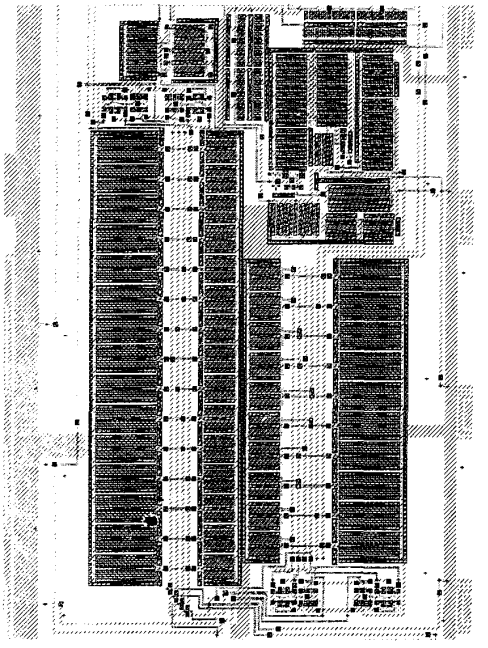

Figure 9: Layout of a SI prototype.

References

[1] S. Haykin: *"Communication Systems"*. Wiley 1978.

[2] W. H. Press et al.: *"Numerical Recipes"*. Cambridge University Press 1986.

[3] R. L. Deraney: *"An introduction to Chaotic Dynamical Systems"*. Benjamin / Cummings 1986.

[4] H.G. Schuster: *"Deterministic Chaos"*. Birkhauser 1984.

[5] R. A. Gregorian and G. C. Temes: *"Analog MOS Integrated Circuits for Signal Processing"*. Wiley 1986

[6] S. Espejo et al.: "Application of Piecewise-Linear Switched-Capacitor Circuits for Random Number Generation". *Proc. Midwest Symp. Circuits Systems*, Aug. 1989.

[7] D.E. Knuth: *"Seminumerical Algorithms"* (2nd edition). Addison Wesley 1981.

[8] J. B. Hughes et al.: "Switched Currents. A new Technique for Analog Sampled-Data Signal Processing". *Proc. International Symp. on Circuits and Systems*, 1989, pp. 1584-1587.